

Data Format Standardization of Space Weather Model Output in the Community Coordinated Modeling Center



QuickTime™ and a
Microsoft Video 1 decompressor
are needed to see this picture.

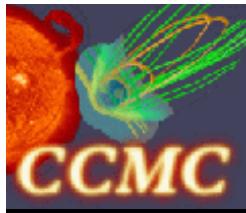
Marlo Maddox

Information Systems Division

Technical Review - Code 587

August 26, 2004

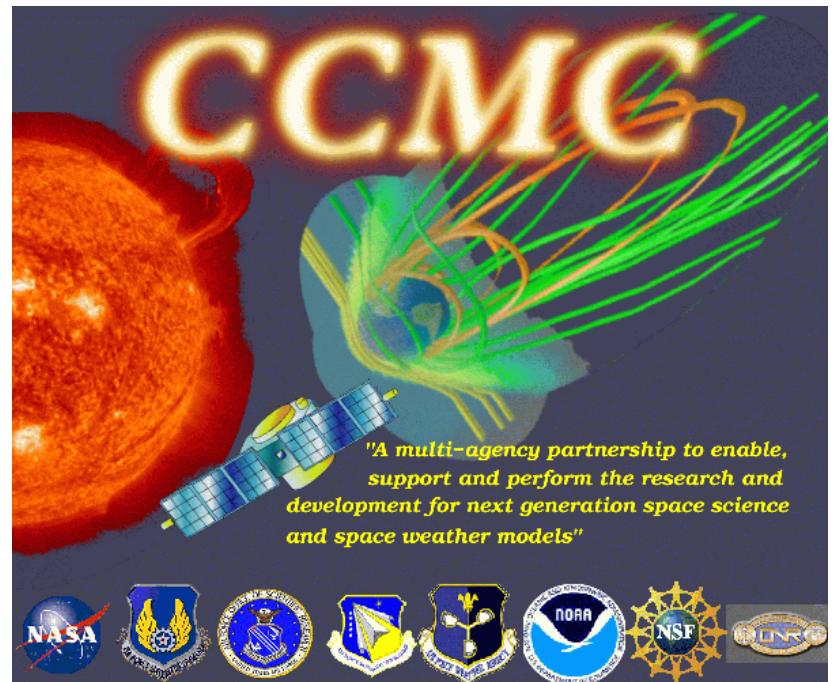




The Community Coordinated Modeling Center

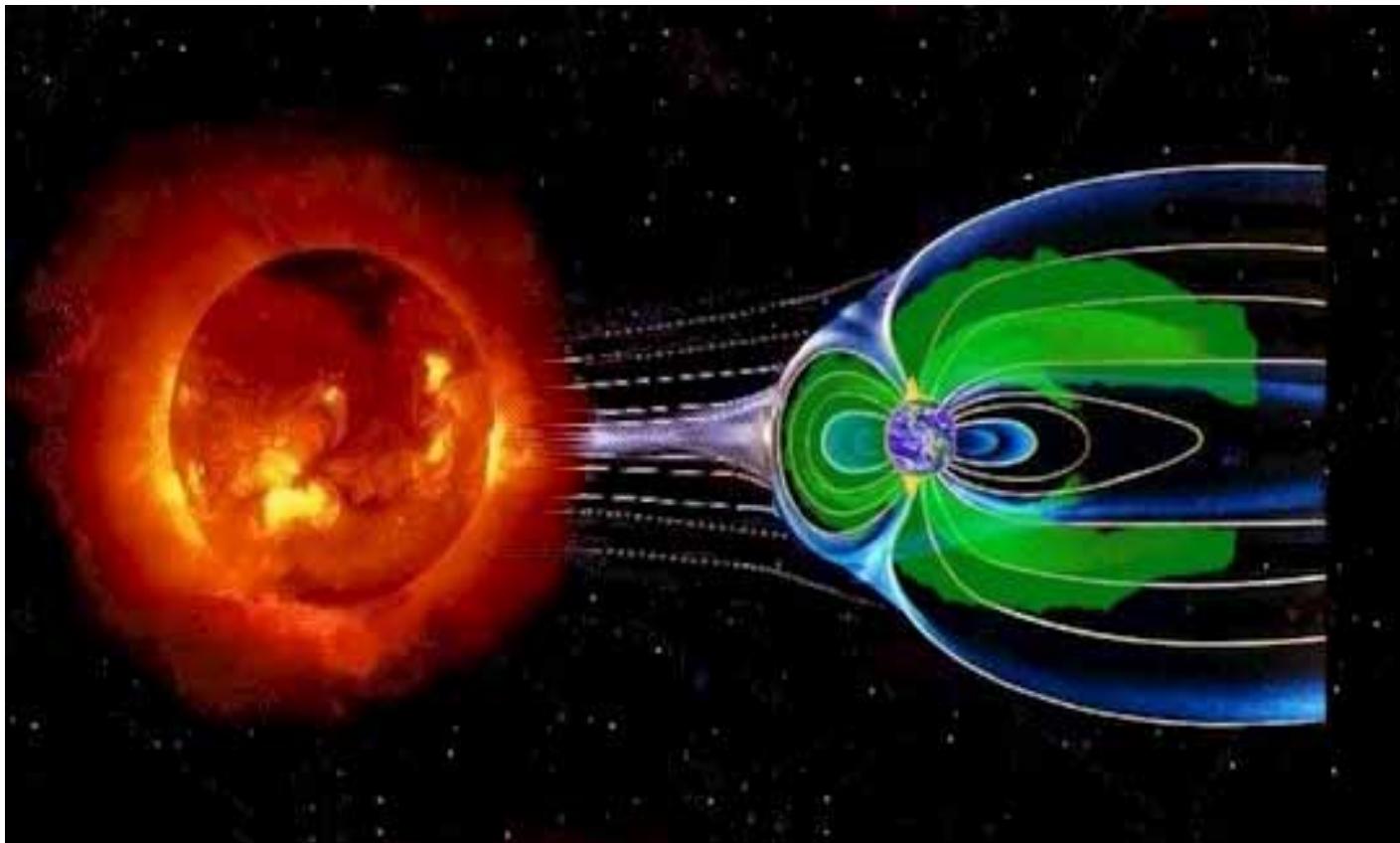
What the CCMC provides:

- Scientific validation
- Model coupling
- Metrics implementations
- Advanced visualization
- Model runs on request



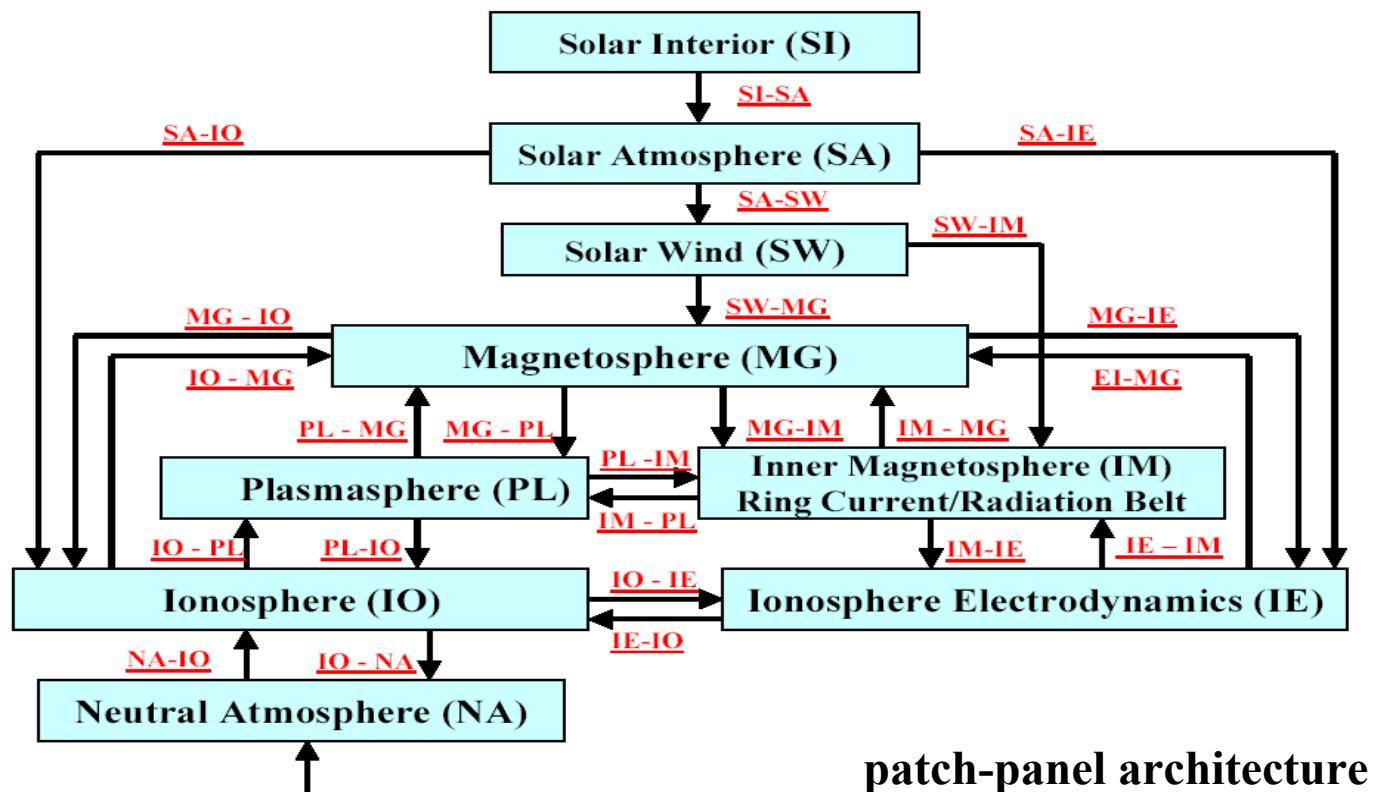


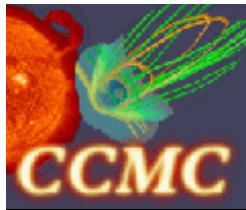
Covering the Entire Domain





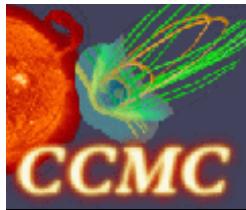
Space Weather Models





Challenges

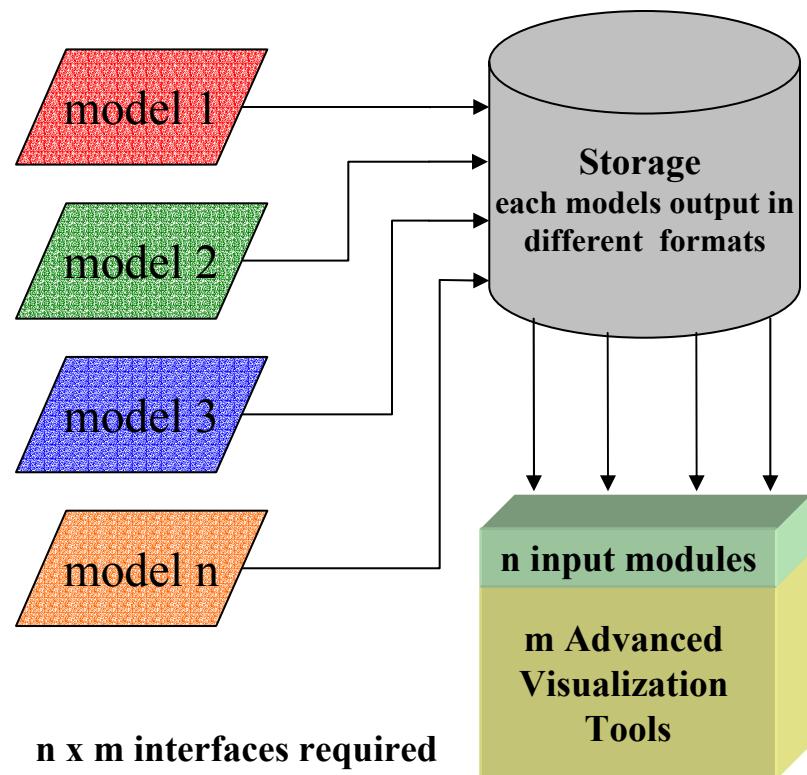
- No rules for standard model interfaces
- Each new model has unique output format
- Developer/user needs to become familiar with internal structure of each output file
- Custom read routines to access model data
- Data is not self describing
- Reduces portability and reuse of
 - Data output itself
 - Tools created to analyze data

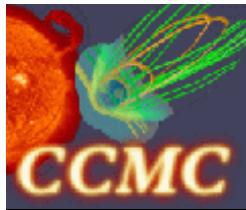


Every Models Output Is Unique

Environment Without Standard

- Specialized I/O routines required for every interface
- Unsuitable for use in flexible model chain
- No commonality between data passing through interfaces

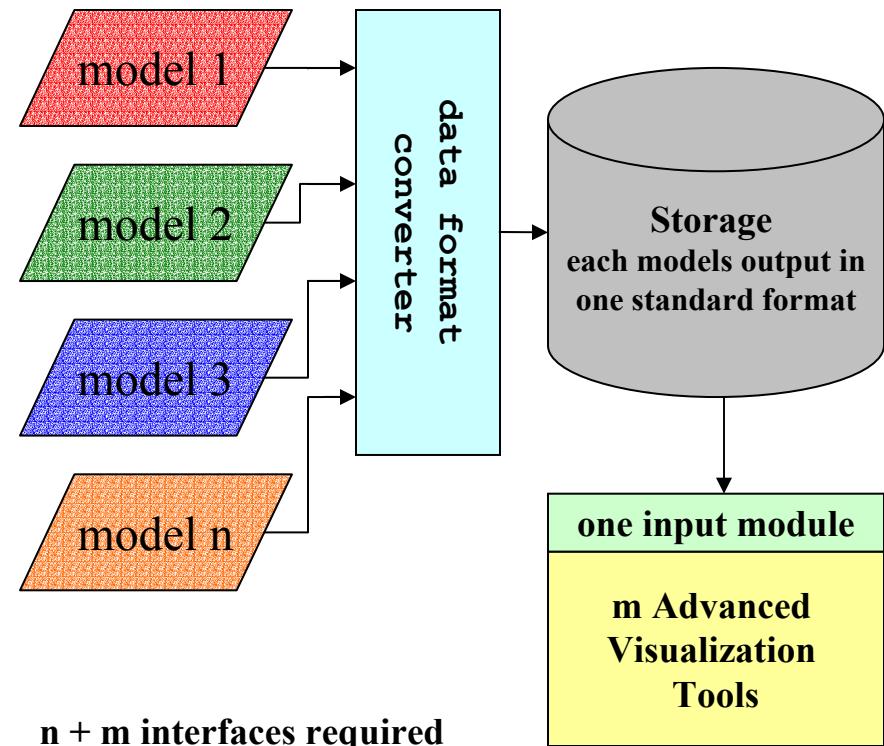


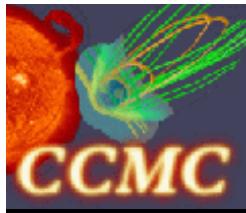


Every Models Output Is Unique

Standardized Environment

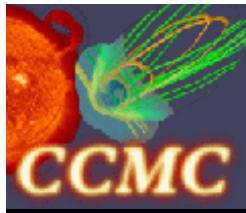
- Original output can be preserved
- Standard format for storage, coupling, & visualization
- Model developers continue to have freedom of choice
- Ensures compatibility between models for coupling
- Ground work for which standard, reusable interfaces and tools can be developed



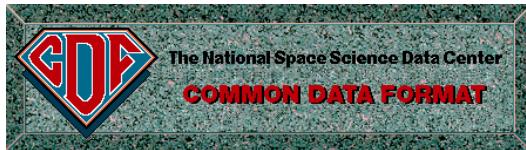


Data Format Standard Options

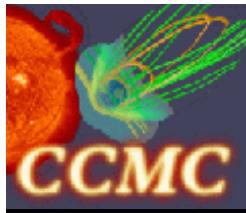
- CDF
- HDF, HDF4, HDF5
- NetCDF
- FITS
- GRIB
- BUFR
- GRADS
- Office Note 29
- Office Note 84
- VICAR
- PDS
- Open Dx Data Model



Data Format Standard Options

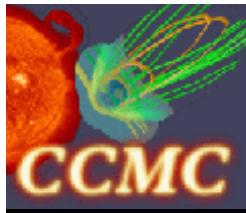


- Developed in 1985 by NSSDC
- Based on conceptual data abstraction where data sets are grouped by variables
- Developed at NCSA at university of Illinois
- Based on hierarchical relationships and dependencies among data



Metadata

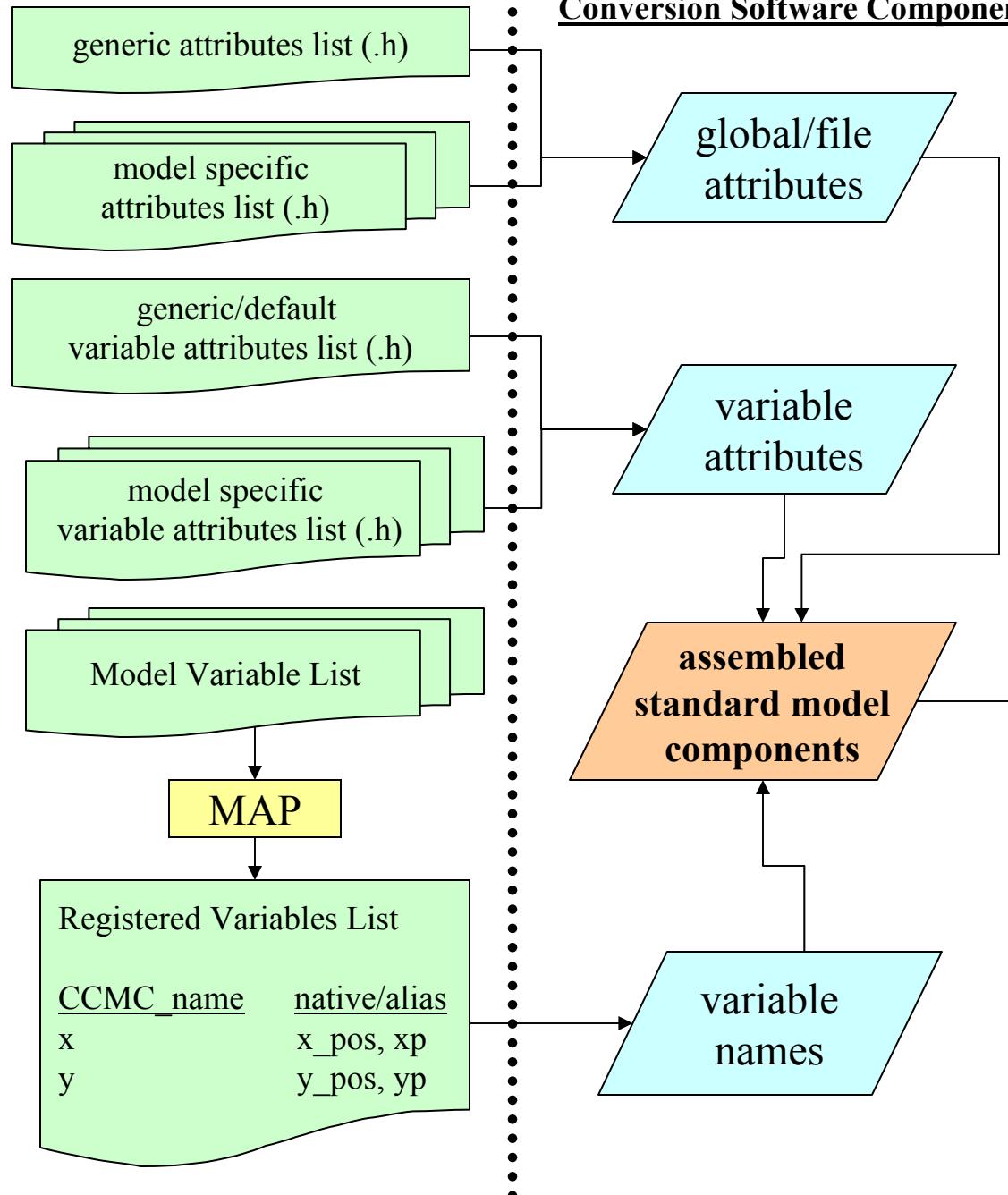
- Aside from the one-to-one data conversion, what additional global & variable meta data do we want to provide?
 - General description of model, howto usage – README
 - Model name and type
 - Date info
 - Run date
 - Generation date
 - Usage
 - Grid Description – # of grids, # of dimensions, dimension size(s)
 - Coordinate system(s)
 - Variable metadata – grid system, min, max, units, description



Software Design Issues

- Identify
 - Core components and modules
 - Interfaces
 - Data Flow paths
- Design should be
 - Flexible
 - Modular
 - Maintenance Friendly

Conversion Software Components



main
conversion
routine

main read driver

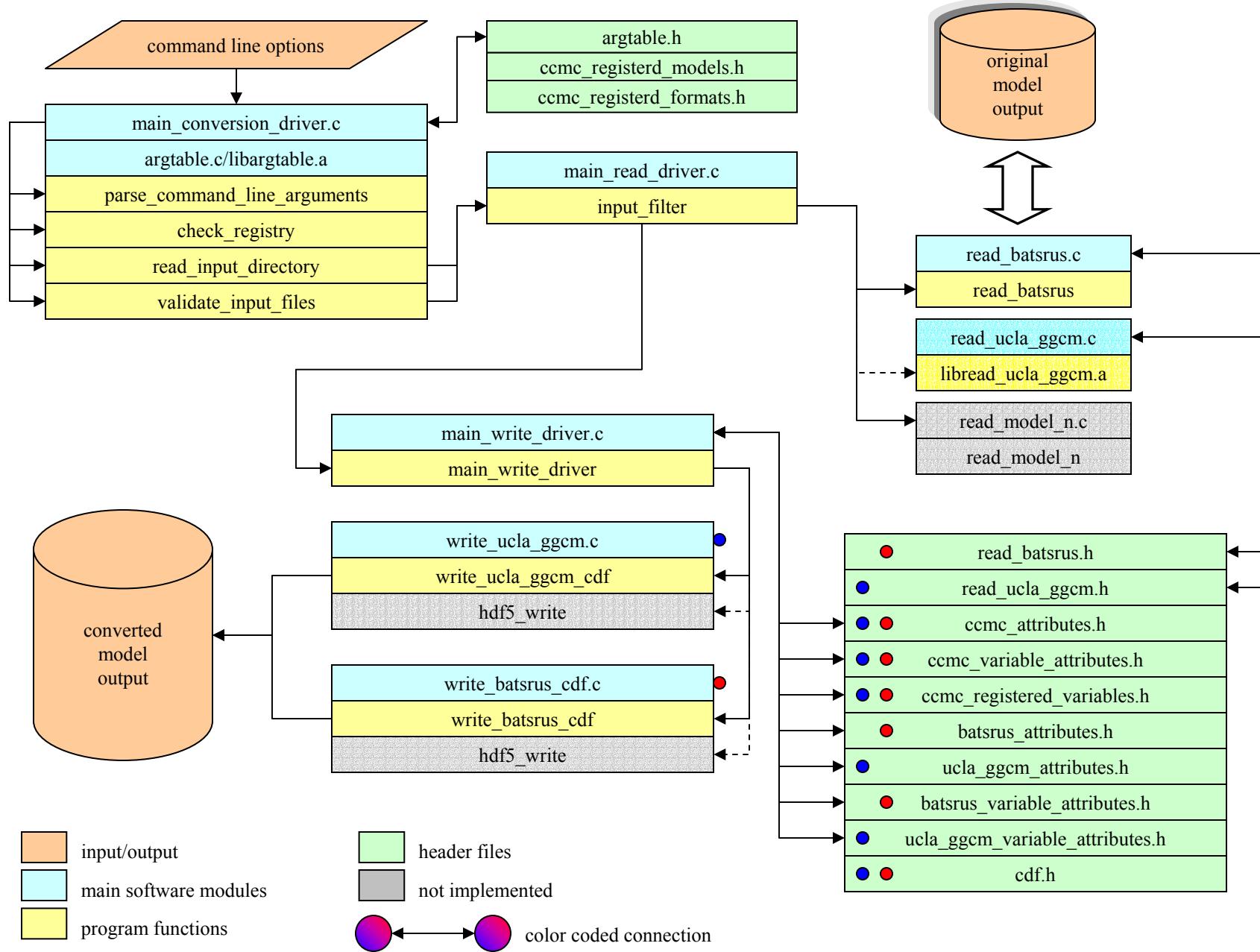
read model a routine
read model b routine
read model n routine

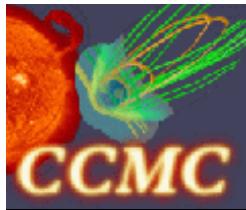
main write driver

convert to cdf
convert to hdf5

standard data file with
common attributes and
variable names for each
registered model

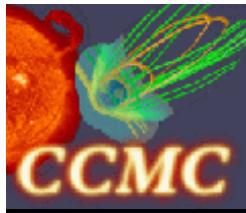
Conversion Software Architecture





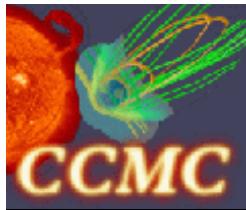
Implementation

- Select model for testing
- Develop & integrate read routine into software
- Develop dynamic interface for metadata & software
- Select optimal CDF storage scheme
- Test Performance
 - file size
 - creation time
 - CDF data access
- Determine feasibility of standardization



Model Selected for Testing

- Block-adaptive-tree-Solarwind-roe-upwind-scheme
(BATSRUS) global magnetosphere MHD model
 - Developed by CSEM at university of Michigan
 - Uses MPI and Fortran 90 standard
 - Executes on massively parallel computer systems
 - Adaptive grid of blocks arranged in varying degrees of spatial refinement levels
 - Solves 3D MHD equations in finite volume form using numerical methods related to roe's approximate Riemann solver
 - Attached to an ionospheric potential solver that provides electric potentials and conductances in the ionosphere

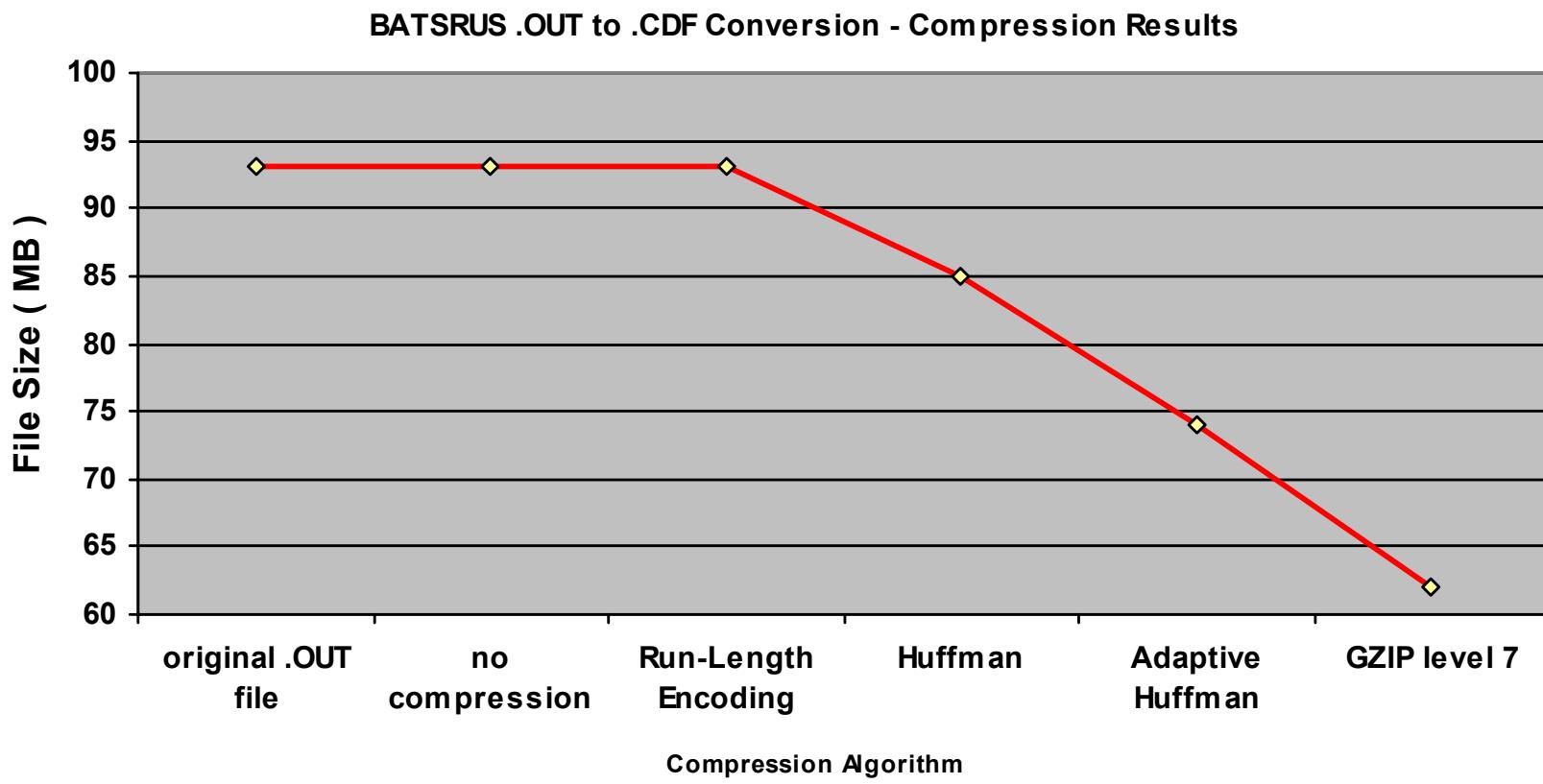


CDF Storage Scheme

- Initial scheme using multiple records for every variable value proved unacceptable
- Current scheme - each BATSRUS variable
 - CDF zVariable
 - Single record
 - Dimension size equal to number of grid positions

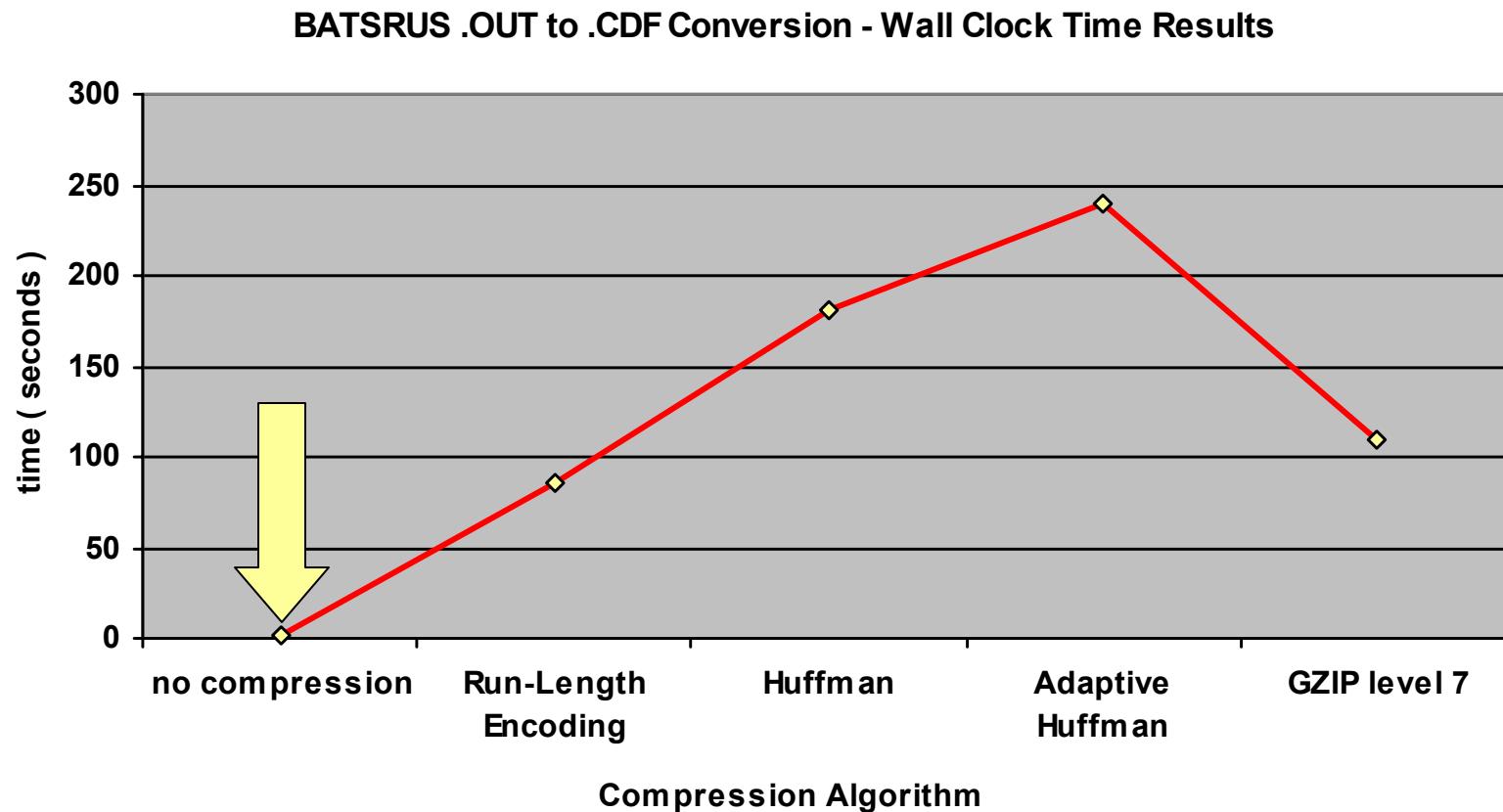


Compression Performance Tests





Compression Performance Tests

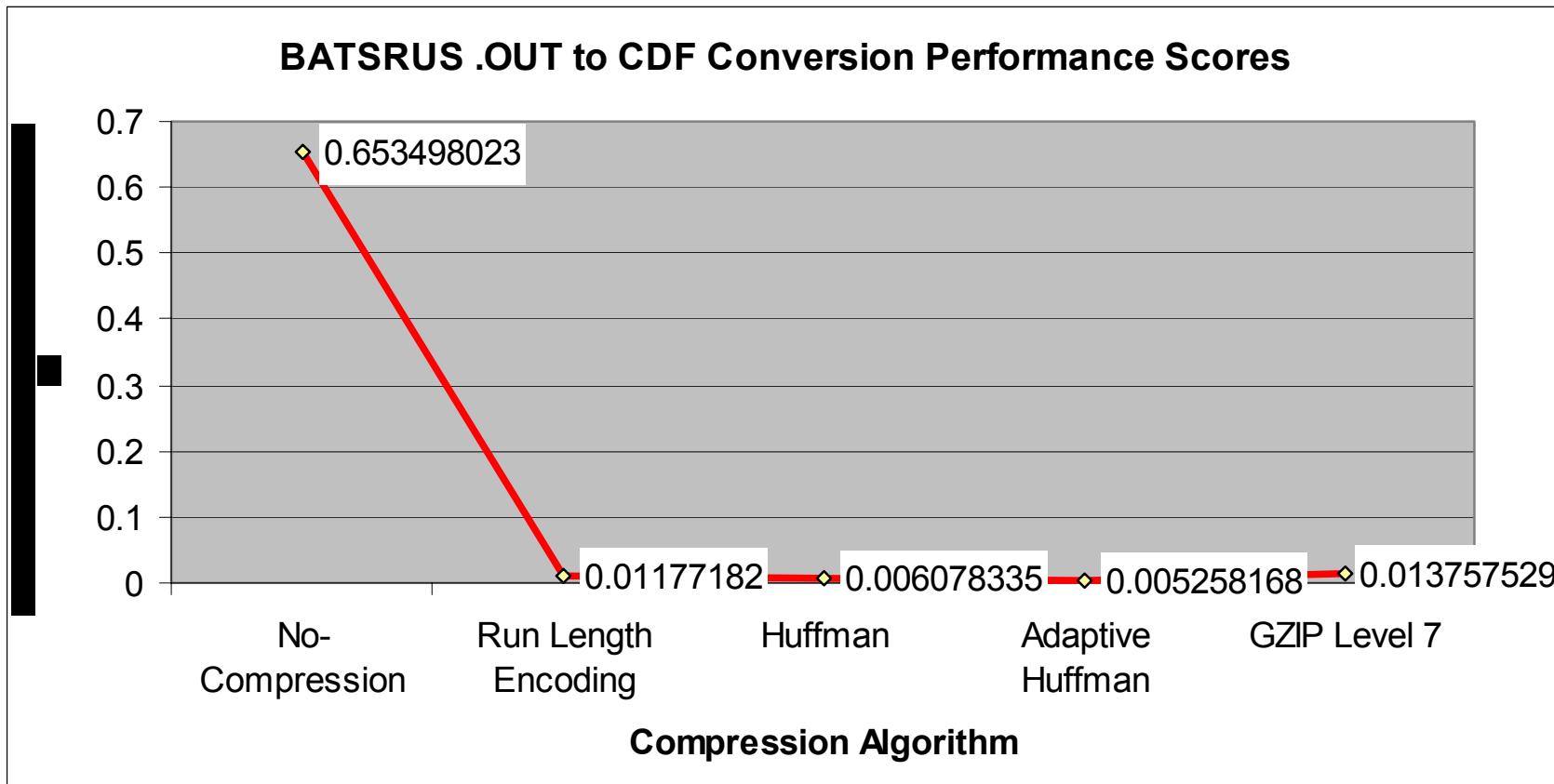




Performance Score

$$\frac{\text{(original_file_size)}}{\text{(cdf_file_size)}} * \frac{1}{t}$$

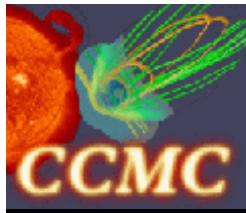
BATSRUS .OUT to CDF Conversion Performance Scores





CDF Data Access

- `CDFvarHyperGet()`
 - Variable name
 - Range of values
 - Sub sampling interval
- `CDFattInquire()`
- `CDFattrGet()`
- `CDFvarInquire()`
- `CDFvarGet()`
- `CDF Toolkit`
 - `skeletontable`
 - `CDFedit`
 - `CDFstats`
 - `CDFexport`
 - `CDFinquire`
 - `CDFcompare`
 - `CDFdir`



CDF Data Access

- Currently creating an user friendly library of routines that can be called from any C supporting language.

```
open_cdf( char *, int number_of_arguments, ... );
interpolate_batsrus_cdf( char *, float, float, float, int, int, ... );
interpolate_ucla_ggcm_cdf( char *, float, float, float, int, int, ... );
close_cdf( void );
get_units( char * );
```

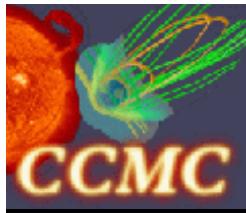
```
open_cdf( cdf_name, 0);
interp_val1 = interpolate_batsrus_cdf( variable1, X, Y, Z, 0, 0 );
close_cdf();
```

```
open_cdf( cdf_name, 3, bx, by, bz);
interp_val1 = interpolate_ucla_ggcm_cdf( variable1, X, Y, Z, 1, 0 );
close_cdf();
```



Foundation is Set

- Initial implementation has proven that data format standardization is feasible and doable
- Discovered optimal internal storage method
- Software design ensures Metadata can be easily modified
- Established a general grid description scheme applicable to any model with any grid & coordinate system
- Use the same process to integrate additional models into the existing software

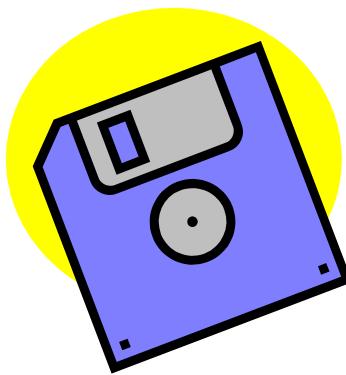


Accomplishments

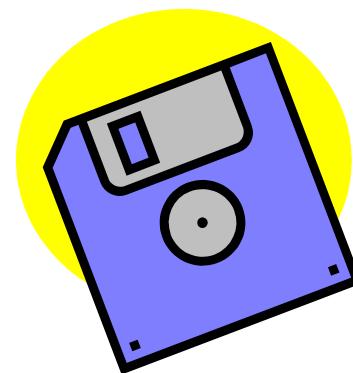
- Clearly defined set of core metadata elements
- Added Block description information to BATSRUS CDF file exploit block structure
 - Required adding 21 additional variables to the 18 existing cdf variables
 - Switched from r to z CDF variables
- Developed a library of routines that interpolate converted BATSRUS CDF data files
- Tested conversion software using 1/16 RE resolution runs
 - 18 million cells
 - 1.2GB sized single time steps
- BATSRUS model support is fully integrated into software
- UCLA-GGCM model support fully integrated into software with interpolation routines also available



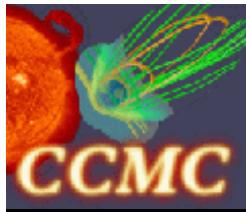
Current CDF Structure



BATSRUS CDF
Skeleton Table



UCLA-GGCM CDF
Skeleton Table

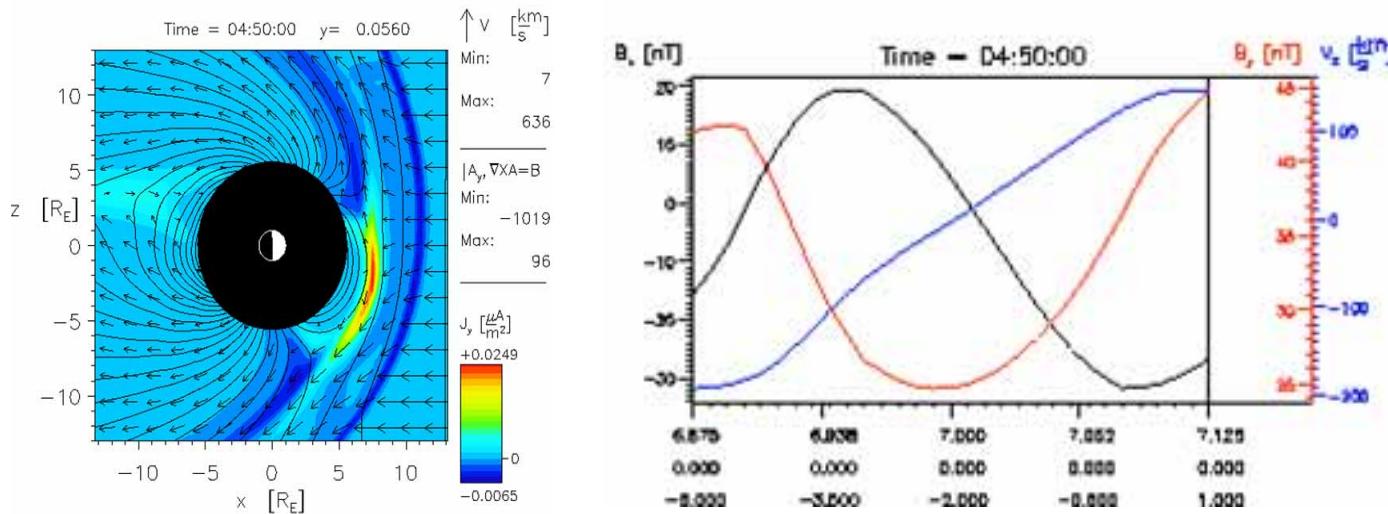


Usage and Benefits

- Speed and efficiency of direct data access
- Easily select individual attributes or variables of interest
- Read subsets of records or single variable values
- Self descriptive files can be used immediately by anyone with CDF tools & libraries
- Same interface regardless of model



Real World Usage



Analysis only
Required
 V_x, B_x, B_y in
Range of $\sim 10 R_E$

2004 Spring AGU Paper

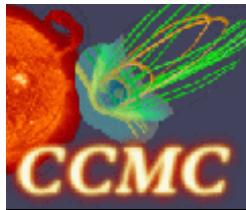
Anti-Parallel Merging vs. Component Dayside Reconnection:
Role in Magnetospheric Dynamics

M M Kuznetsova, M Hesse, L Rastaetter, M. Maddox, D De Zeeuw, T I Gombosi



Real World Usage

- Particle Tracing Research using BATSRUS MHD Data for given time & position:
 - B_x, B_y, B_z
 - V_x, V_y, V_z
- IDL & OpenDx visualization routines are Migrating to CCMC CDF files



Outstanding Issues

- Variable naming conventions
- How much metadata to pack into each file
- Grid description
- Coordinate system transformations
- Making the transition
- As Simulations grow larger keeping original model output may not be possible



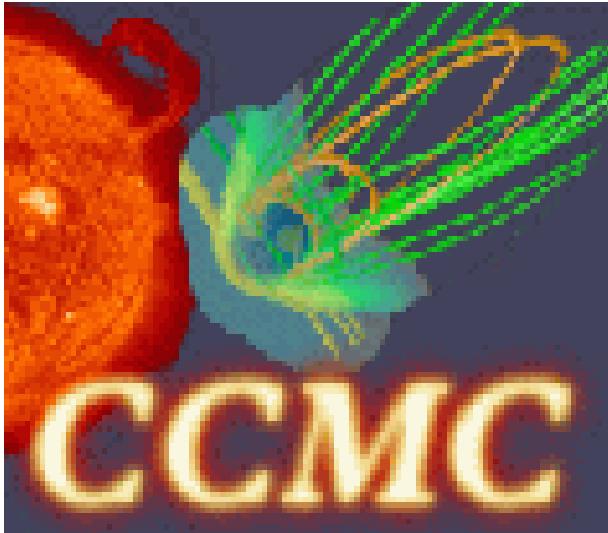
Next Steps

- Refine standard grid description scheme
- Implement HDF 5 conversion module
- Test output conversion performance with HDF 5 data standard
- Compare CDF vs. HDF 5 performance
- Decide use of either CDF or HDF5 or both



Acronym List

3D	3 Dimensional
AGU	American Geophysical Union
amu	Atomic Mass Unit
ASCII	American Standard for Information Interchange
BATSRUS	Block-Adaptive-Tree-Solar-Wind-Roe_Upwind-Scheme
CCMC	Community Coordinated Modeling Center
CDF	Common Data Format
CDHF	Central Data Handling Facility
CSEM	Center for Space Environment Modeling
GSFC	Goddard Space Flight Center
GZIP	GNU ZIP
HDF 5	Hierarchical Data Format 5
I/O	Input/Output
MHD	Magnetohydro Dynamics
MPI	Message Passing Interface
NASA	National Aeronautics and Space Administration
NCSA	National Center for Super Computing Applications
NSSDC	National Space Science Data Center
Open Dx	Open Data Explorer
PERL	Practical Extraction and Reporting Language
RE	Earth Radius
STDOUT	Standard Output
UCLA	University of California, Los Angeles
UCLA-GGCM	UCLA Geospace General Circulation Model

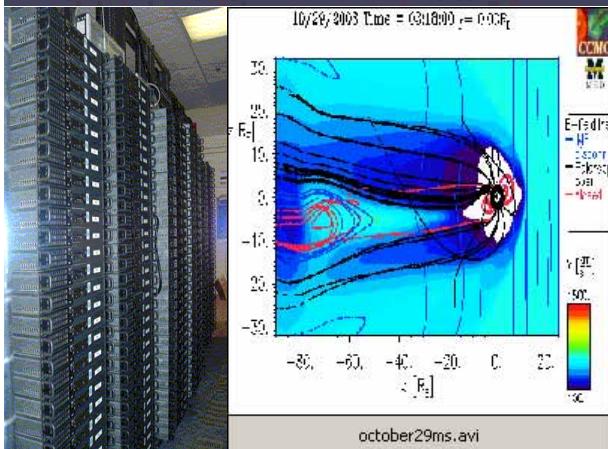


Data Format Standardization of Space Weather Model Output in the Community Coordinated Modeling Center

Marlo Maddox

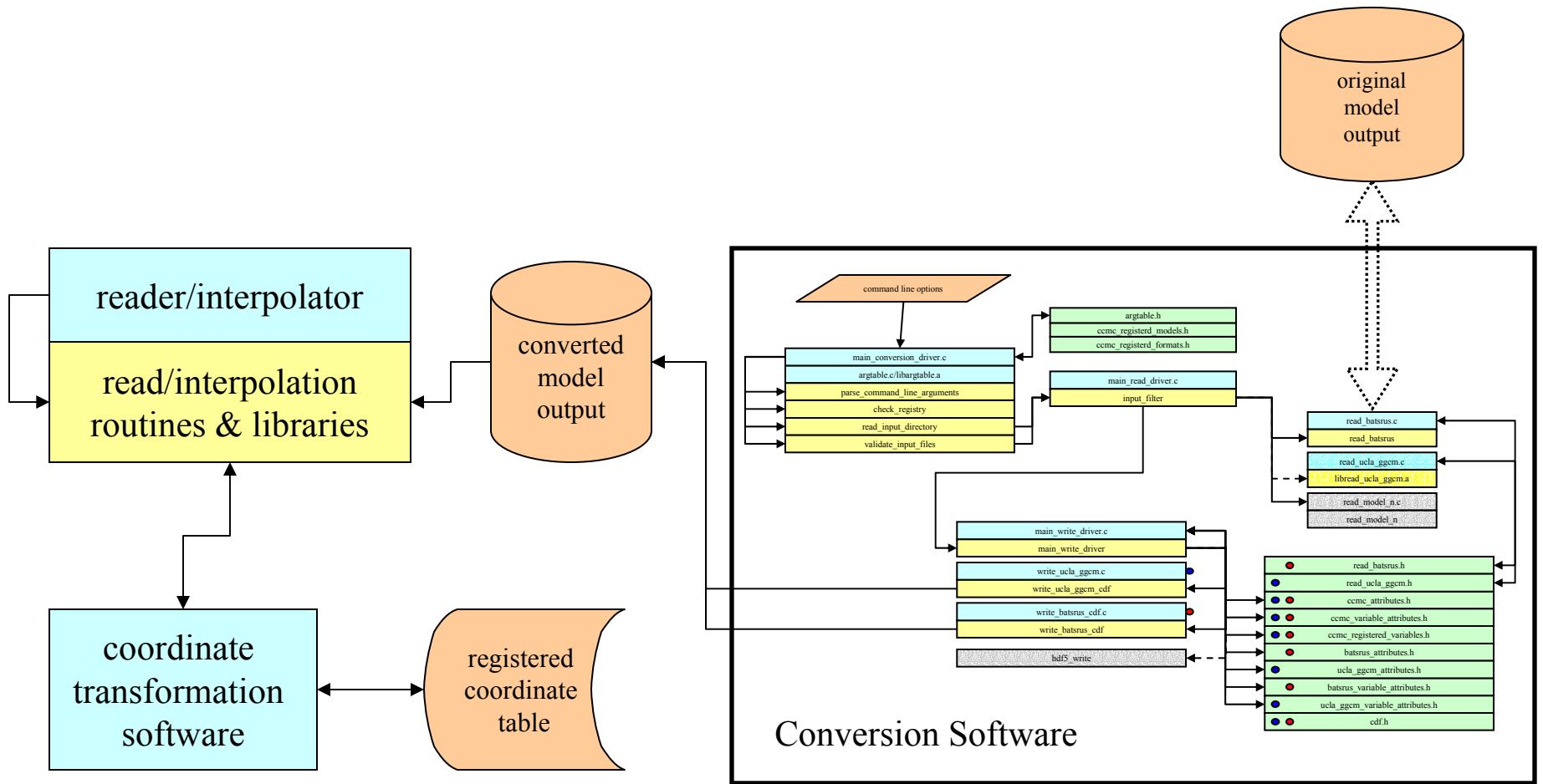
Laboratory for Extraterrestrial Physics
Sun-Earth Connections Seminar

June 4, 2004



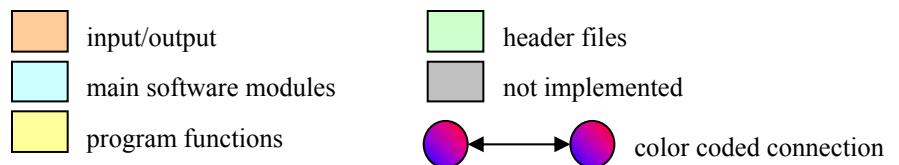


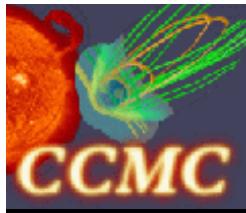
Supplemental Sides



reader/interpolator – built on top of read/interpolation libraries and routines which themselves can be called directly from an external program.

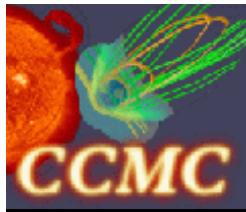
coordinate transformation software – called to convert a user supplied coordinate system to the system in which the model data is currently stored.





Model Selected for Testing

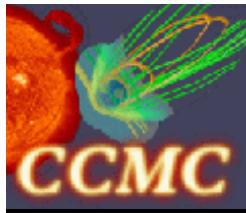
- Block-adaptive-tree-Solarwind-roe-upwind-scheme
(BATSRUS) global magnetosphere MHD model
 - Developed by CSEM at university of Michigan
 - Uses MPI and Fortran 90 standard
 - Executes on massively parallel computer systems
 - Adaptive grid of blocks arranged in varying degrees of spatial refinement levels
 - Solves 3D MHD equations in finite volume form using numerical methods related to roe's approximate Riemann solver
 - Attached to an ionospheric potential solver that provides electric potentials and conductances in the ionosphere



Understanding the BATSRUS Models Output

General Scientific Output

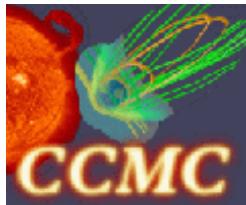
- magnetospheric plasma parameters
 - Atomic mass unit density
 - Pressure
 - Velocity
 - Magnetic field
 - Electric currents
- ionospheric parameters
 - Electric potential
 - Hall and Pedersen conductances



Understanding the BATSRUS Models Output

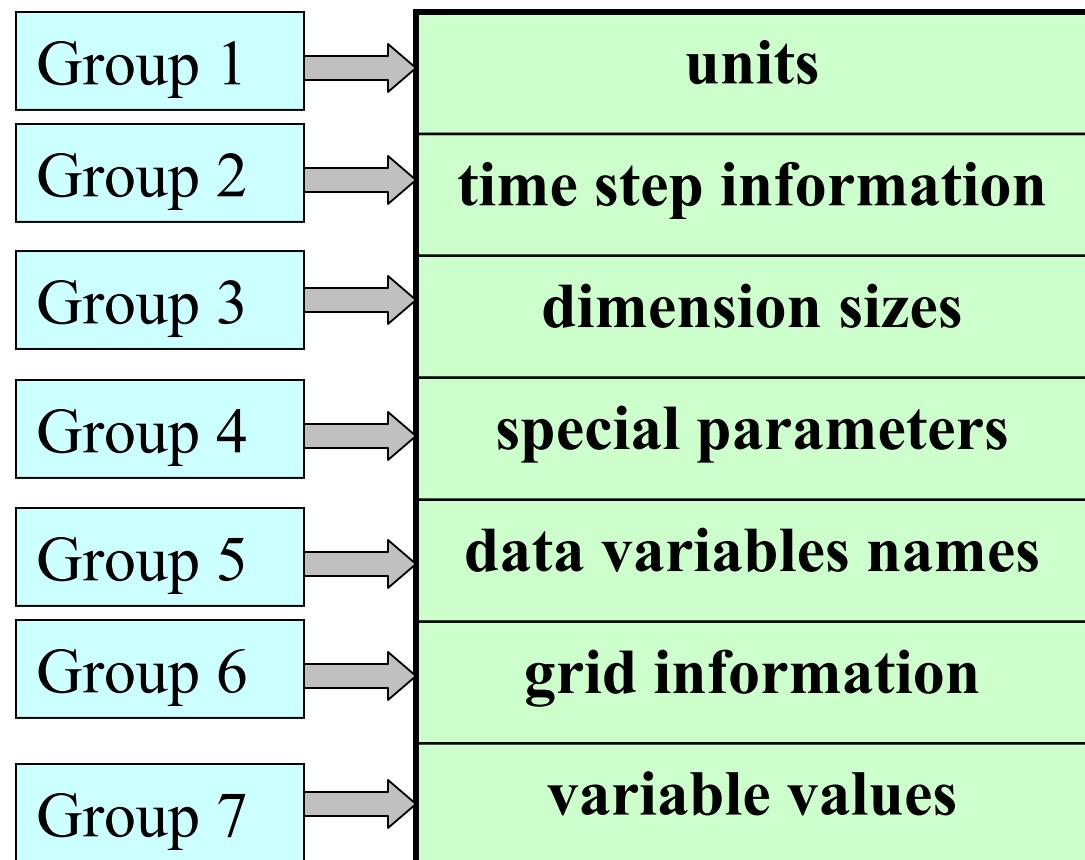
Core Output Files

- **start_time** – ascii text file listing the beginning of the models execution time
- **3d_ful_1_list** – ascii text file listing the multiple output file names and their corresponding time steps
- ***.h** –ascii text file for each time step, lists general information about data for current iteration
- ***.OUT -binary file for each time step, contains all data for current iteration**



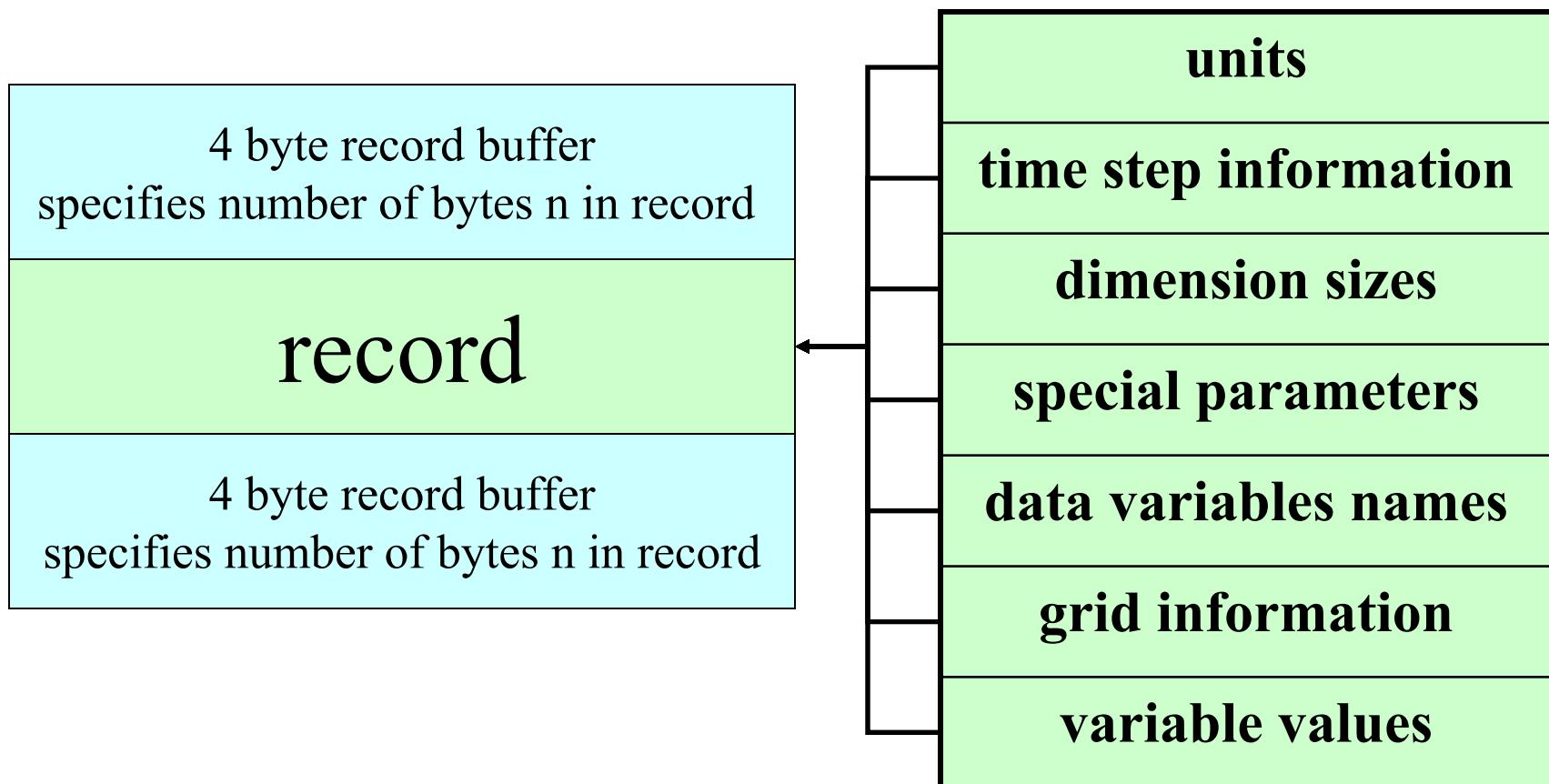
BATSRUS .OUT File

The BATSRUS .OUT file consists of 7 top level groups which themselves contain a number of individual records





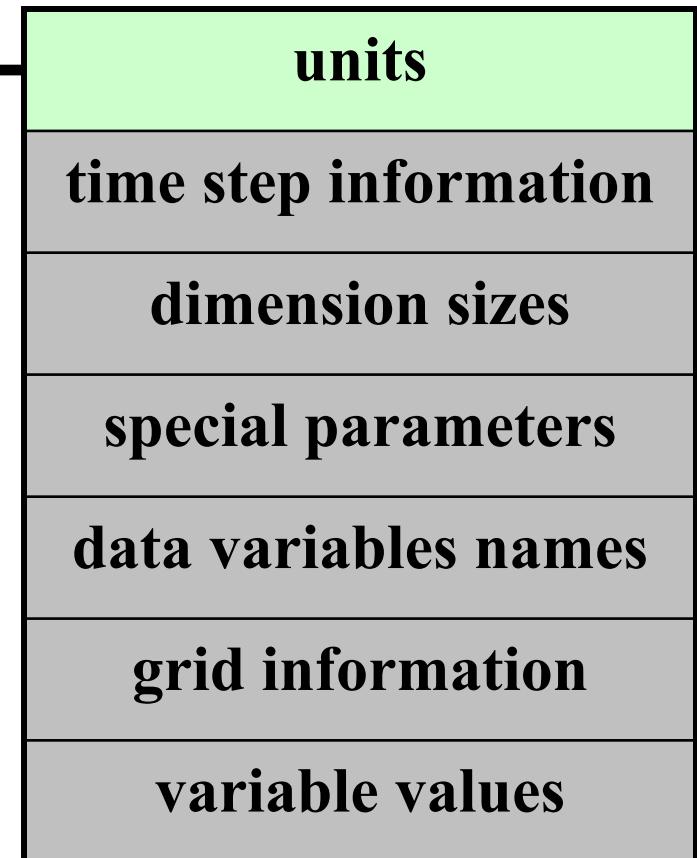
BATSRUS .OUT File





BATSRUS .OUT File

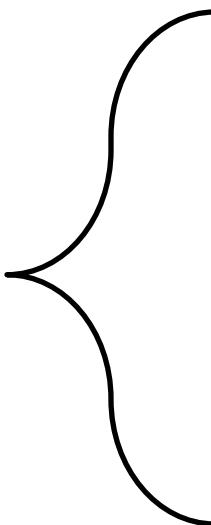
byte	value
1	
2	
3	
4	
5	n bytes containing units for variables R amu/cm ³ km/s nT nPa J/m ³ uA/m ²
n	
n+1	
n+2	
n+3	
n+4	number of bytes n for previous record



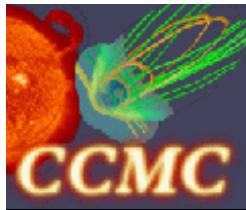


BATSRUS .OUT File

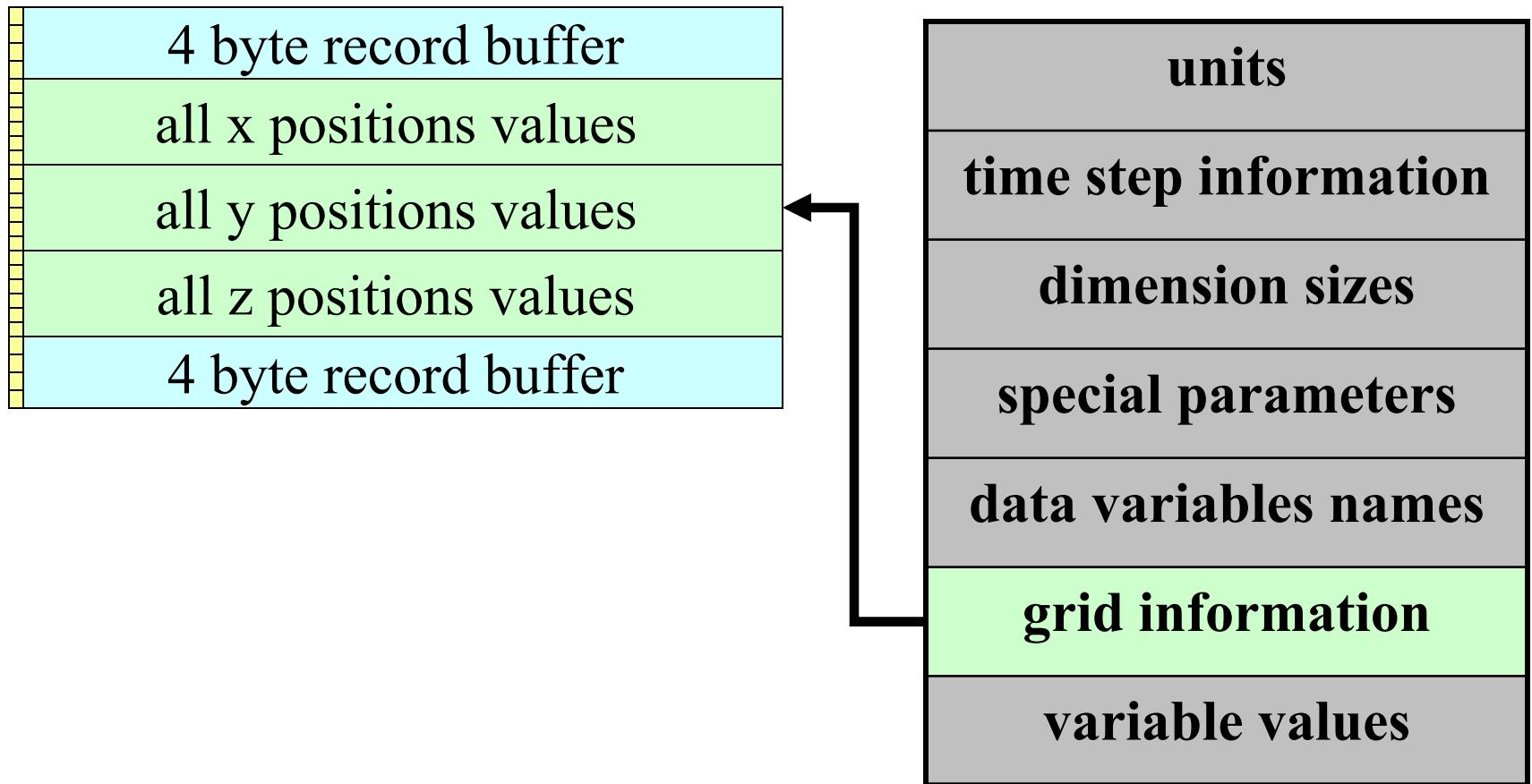
- general information
- static non-variant data



units
time step information
dimension sizes
special parameters
data variables names
grid information
variable values

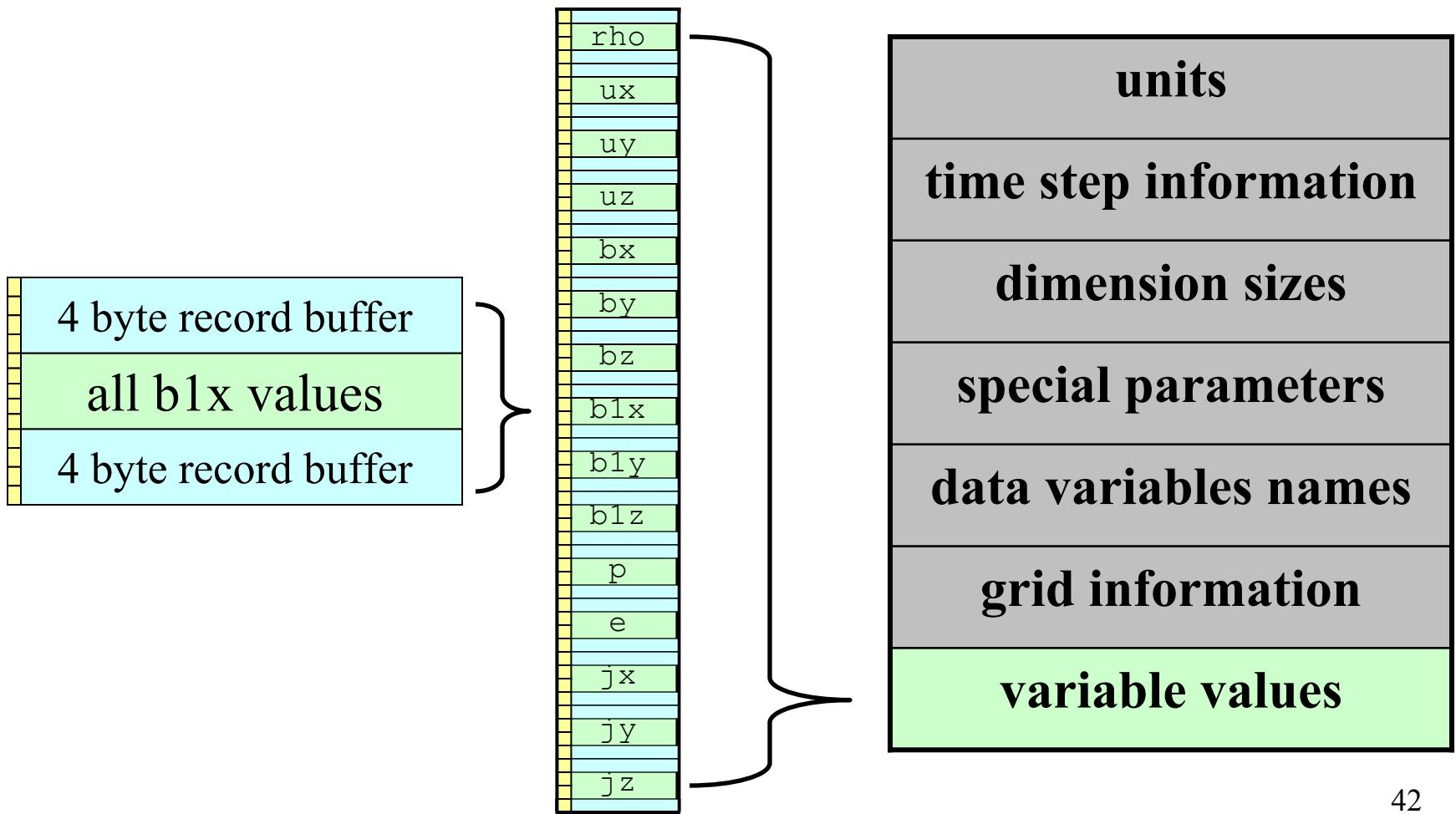


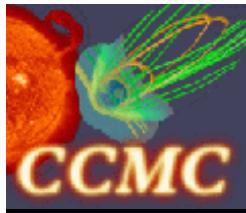
BATSRUS .OUT File





BATSRUS .OUT File

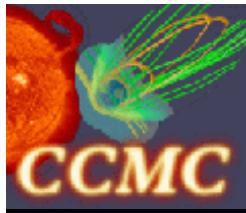




Core BATSRUS Data

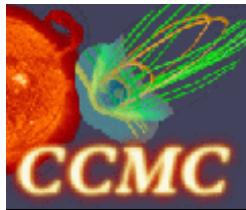
Sample Position & Plot Data from Read Routine

[x0,y0,z0]	[x1,y1,z1]	[x2,y2,z2]
[-251,-44,4]	[-243,-44,4]	[-235,-44,4]
rho = 35.0913009643555 amu/cm3	rho = 34.8759002685547 amu/cm3	rho = 34.5895004272461 amu/cm3
ux = -425.023010253906 km/s	ux = -423.704986572266 km/s	ux = -421.550994873047 km/s
uy = -0.541957974433899 km/s	uy = -1.95363998413086 km/s	uy = -2.99478006362915 km/s
uz = 61.4497985839844 km/s	uz = 64.7194976806641 km/s	uz = 67.2033004760742 km/s
bx = -0.04436020180583 nT	bx = 0.0458825007081032 nT	bx = 0.157462000846863 nT
by = 18.2842998504639 nT	by = 18.6569004058838 nT	by = 18.8631000518799 nT
bz = -0.25143301486969 nT	bz = -0.75831001996994 nT	bz = -1.14868998527527 nT
blx = -0.0431502014398575 nT	blx = 0.0472049005329609 nT	blx = 0.158911004662514 nT
bly = 18.2845993041992 nT	bly = 18.6573009490967 nT	bly = 18.8635005950928 nT
blz = -0.253215998411179 nT	blz = -0.760269999504089 nT	blz = -1.15085005760193 nT
p = 0.108653999865055 nPa	p = 0.105824999511242 nPa	p = 0.102815002202988 nPa
e = 5.70820013479079e-09 J/m3	e = 5.6559201766504e-09 J/m3	e = 5.56774981674835e-09 J/m3
jx = 4.19913994846866e-05 uA/m2	jx = 4.3270400055917e-05 uA/m2	jx = 4.39381983596832e-05 uA/m2
jy = 7.8943401149445e-07 uA/m2	jy = 3.92857009501313e-06 uA/m2	jy = 3.08662993120379e-06 uA/m2
jz = 1.26951999845915e-05 uA/m2	jz = 1.348850037175e-05 uA/m2	jz = 9.81443008640781e-06 uA/m2



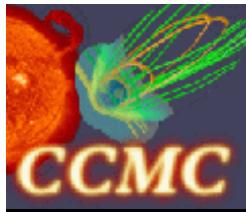
CCMC CDF Global Attributes

- Created new global attributes
 - Model Name
 - Model Type
 - Generation date
 - Original Filename
 - Run registration Number
 - Generated By
 - Terms of Usage
- Incorporated existing data from groups 1 - 5
 - units**
 - time step information**
 - dimension sizes**
 - special parameters**
 - data variables names**
- Incorporate attributes for grid description



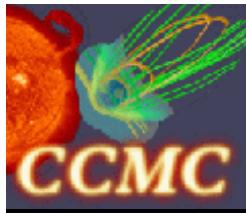
CDF Variables

- CDFs contain two types of variables
 - rVariables – all have the same dimensionality
 - zVariables – can each have different dimensionalities
- CDF Dimensionality
 - a variable with one dimension is like an array
 - number of elements in array correspond to the dimension size

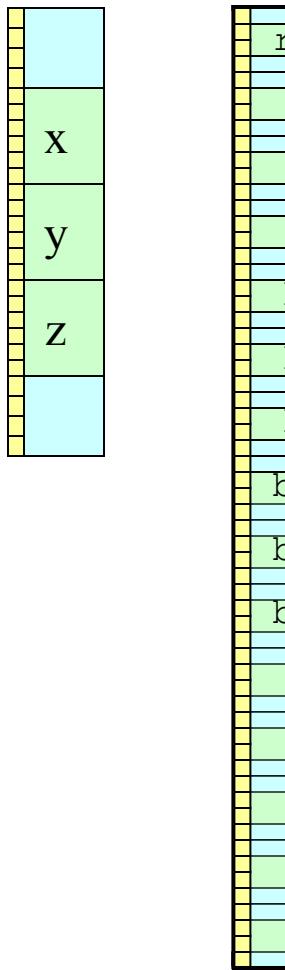


CDF Storage Scheme

- First scheme using multiple records for every variable value proved unacceptable
- Store each BATSRUS variable
 - CDF rVariable
 - Single record
 - Dimension size equal to number of grid positions



CCMC CDF Variables



- BATSRUS model contains 18 dynamic variables
 - 3 position variables
 - 15 plot variables
- 18 CDF rVariables
 - one record per variable
 - one dimensional variables
 - dimension size = number of cells in grid
 - 18 records vs. 10.4 million in first storage scheme





BATRUS .OUT to CDF

x variable

1:[1] = -251.0

1:[2] = -243.0

1:[3] = -235.0

1:[4] = -227.0

1:[5] = -219.0

1:[6] = -211.0

1:[7] = -251.0

1:[8] = -243.0

1:[9] = -235.0

1:[10] = -227.0

1:[11] = -219.0

1:[12] = -211.0

1:[13] = -251.0

1:[14] = -243.0

1:[15] = -235.0

1:[16] = -227.0

1:[17] = -219.0

1:[18] = -211.0

1:[19] = -251.0

1:[20] = -243.0

1:[21] = -235.0

1:[22] = -227.0

1:[23] = -219.0

1:[24] = -211.0

1:[1283401] = -251.0

1:[1283402] = -243.0

1:[1283403] = -235.0

1:[1283404] = -227.0

1:[1283405] = -219.0

1:[1283406] = -211.0

1:[1283407] = -251.0

1:[1283408] = -243.0





BATRUS .OUT to CDF

first column indicates current record number

column two references the current records element index – each element of the record stores a value for the current variable

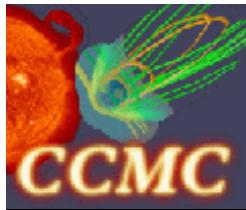
1:[1] = -251.0
1:[2] = -243.0
1:[3] = -235.0
1:[4] = -227.0
1:[5] = -219.0
1:[6] = -211.0
1:[7] = -251.0
1:[8] = -243.0

1:[9] = -235.0
1:[10] = -227.0
1:[11] = -219.0
1:[12] = -211.0
1:[13] = -251.0
1:[14] = -243.0
1:[15] = -235.0
1:[16] = -227.0

1:[17] = -219.0
1:[18] = -211.0
1:[19] = -251.0
1:[20] = -243.0
1:[21] = -235.0
1:[22] = -227.0
1:[23] = -219.0
1:[24] = -211.0

1:[1283401] = -251.0
1:[1283402] = -243.0
1:[1283403] = -235.0
1:[1283404] = -227.0
1:[1283405] = -219.0
1:[1283406] = -211.0
1:[1283407] = -251.0
1:[1283408] = -243.0





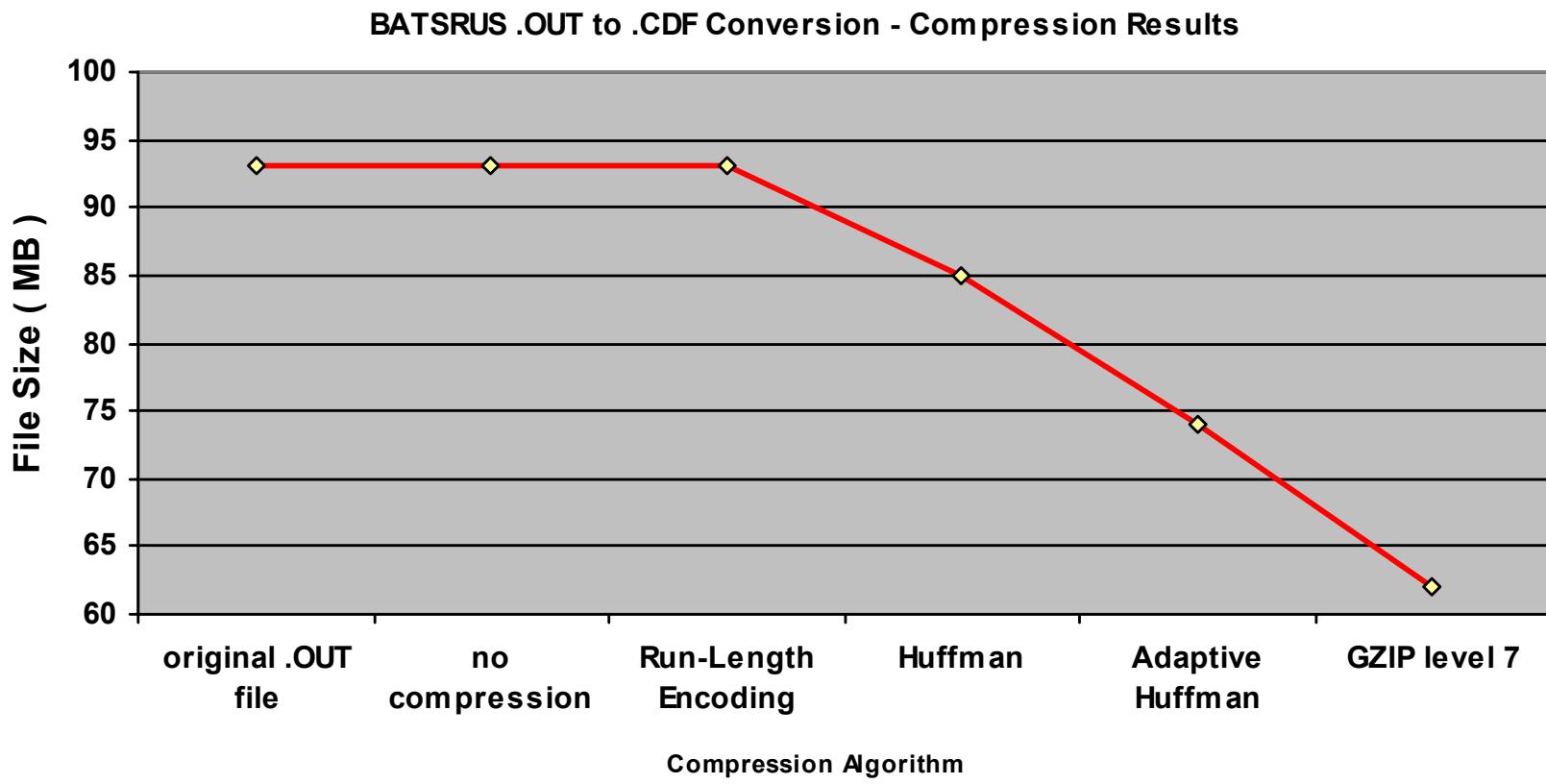
BATRUS .OUT to CDF

- Original BATRUS .OUT file
 - 92 MB
 - 1,293,408 grid positions
- BATRUS CDF Output File
 - 92 MB
 - 18 records
 - 18 rVariables
 - one dimensional
 - 1.3 million elements



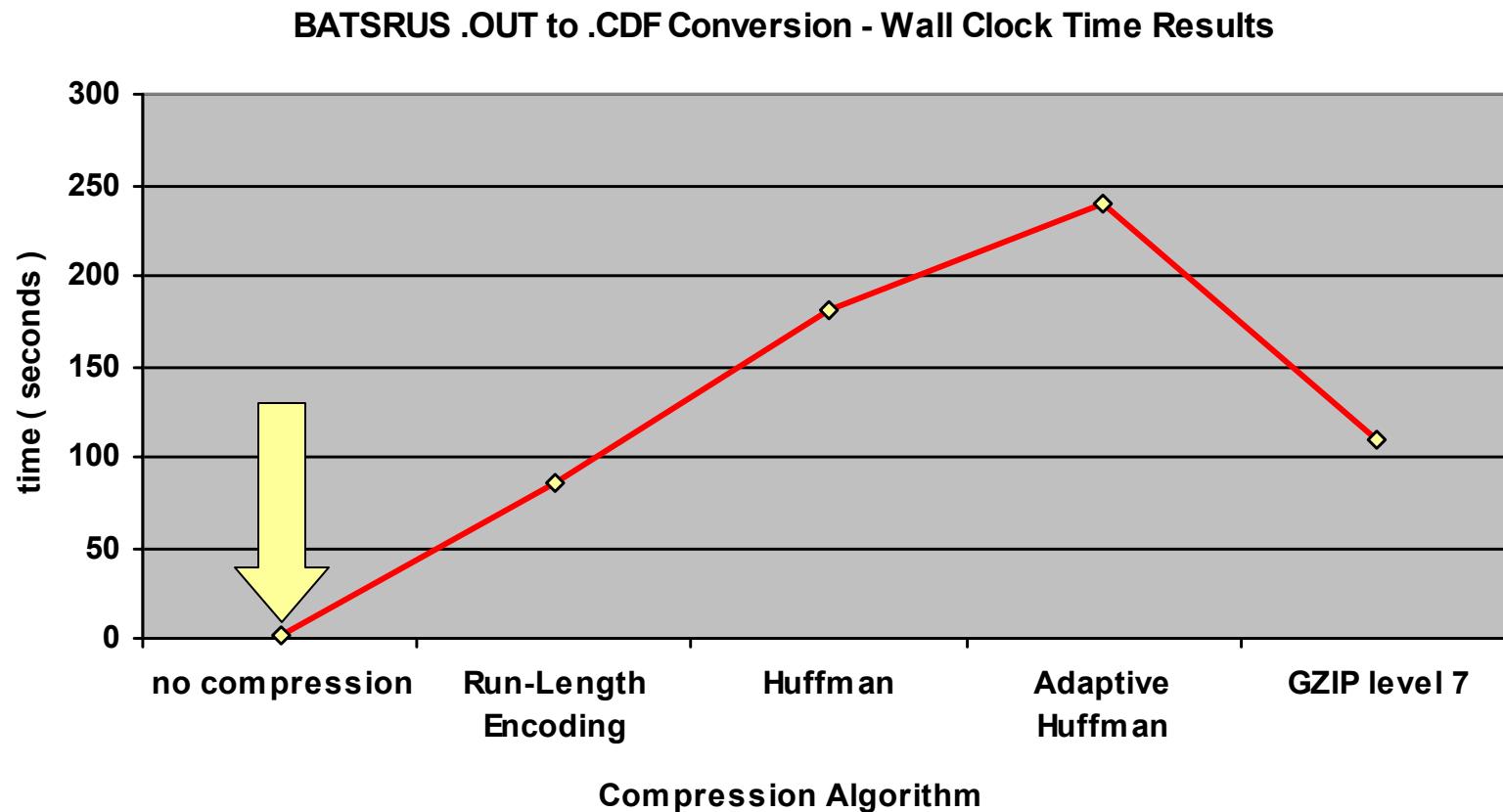


Compression Performance Tests





Compression Performance Tests

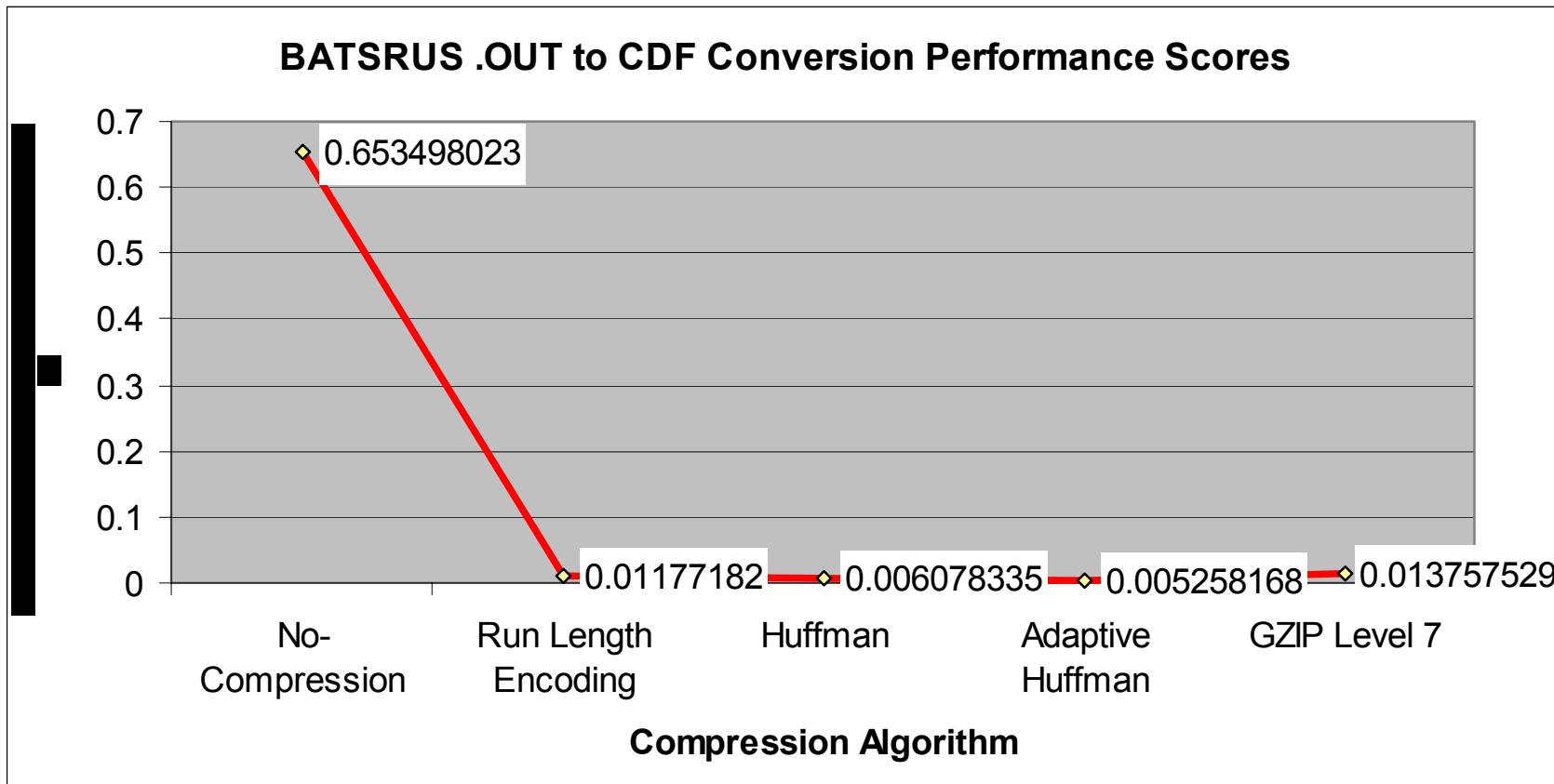




Performance Score

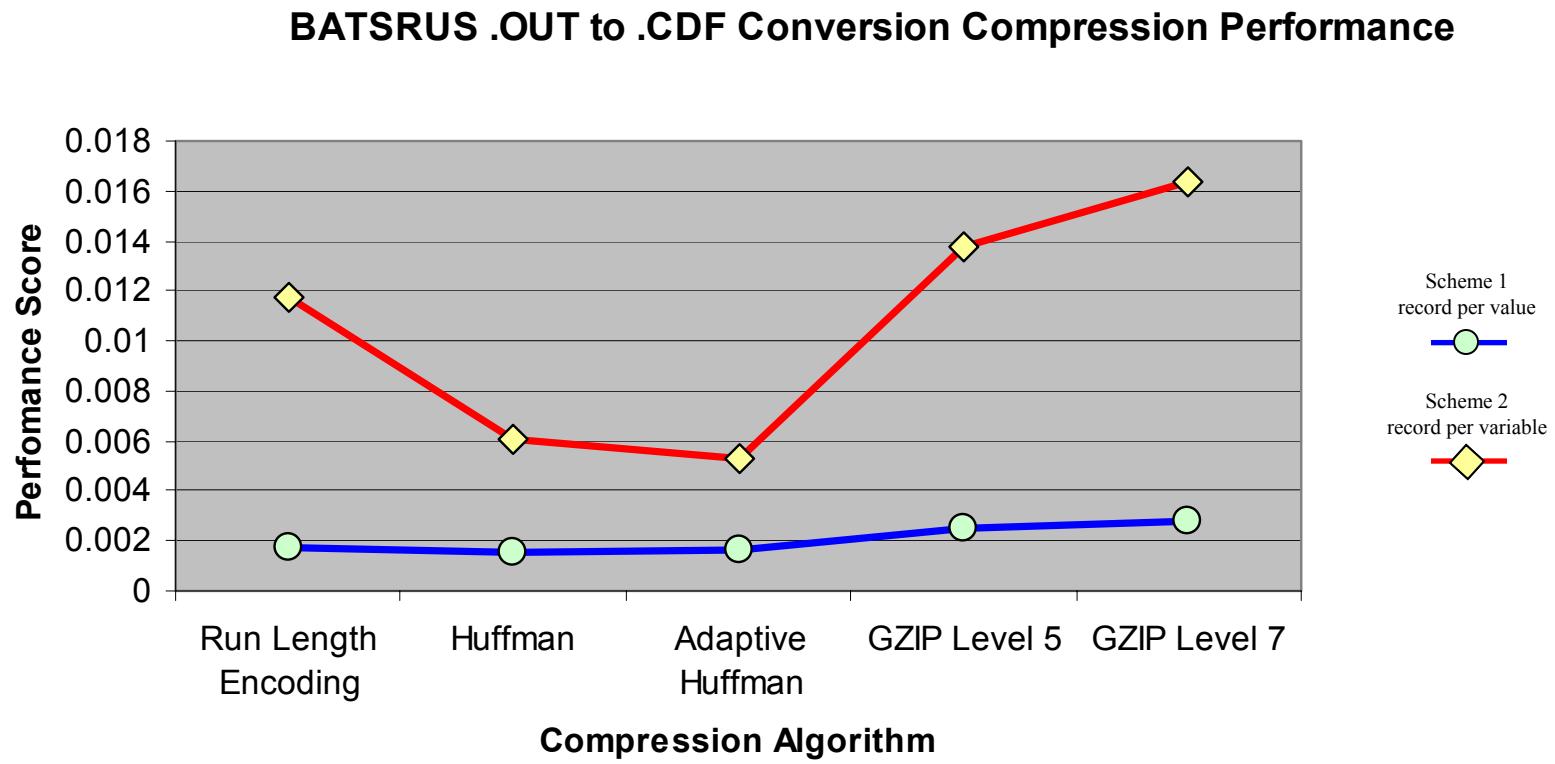
$$\frac{\text{(original_file_size)}}{\text{(cdf_file_size)}} * \frac{1}{t}$$

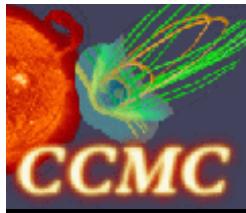
BATSRUS .OUT to CDF Conversion Performance Scores





Compression Performance Comparison





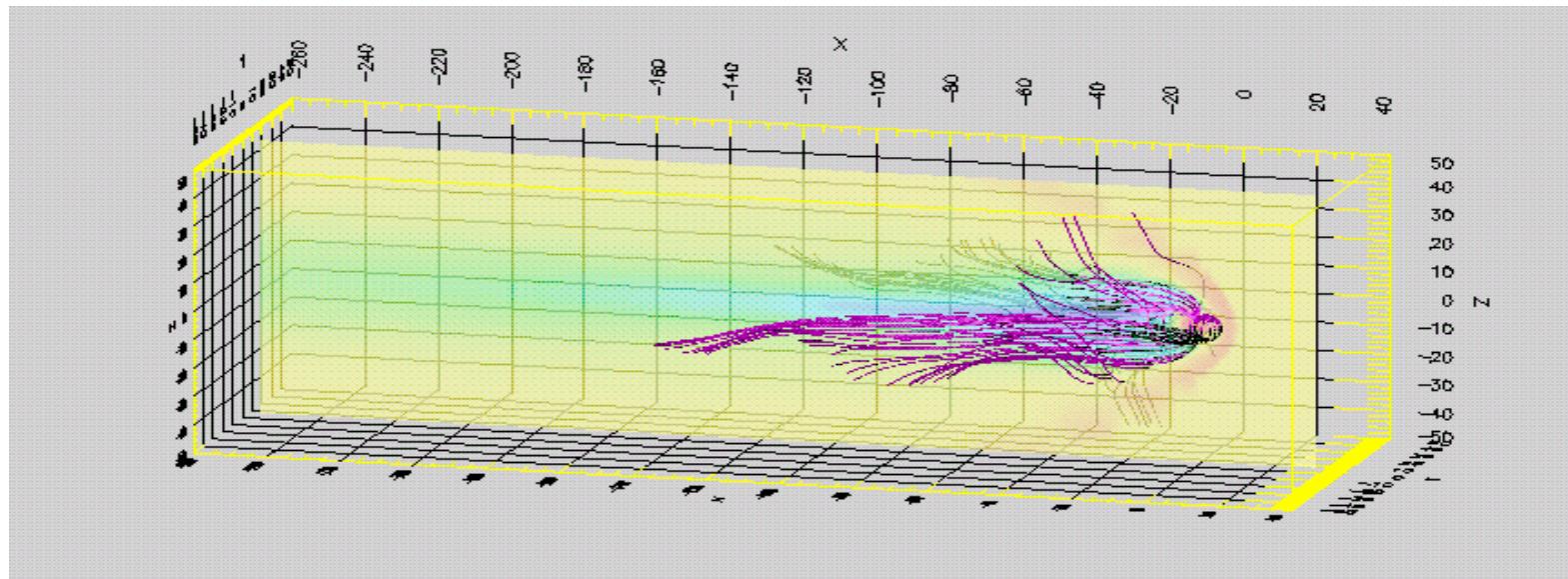
CDF Conversion Performance Results

- Optimal CDF storage format
 - Single one-record rVariables
 - Dimension size equal to number of cells in grid
- Uncompressed CDF creation time of 1.5 seconds
- CDF file size virtually the same as original BATSRUS output file size
- Method could be applied to additional models in similar fashion



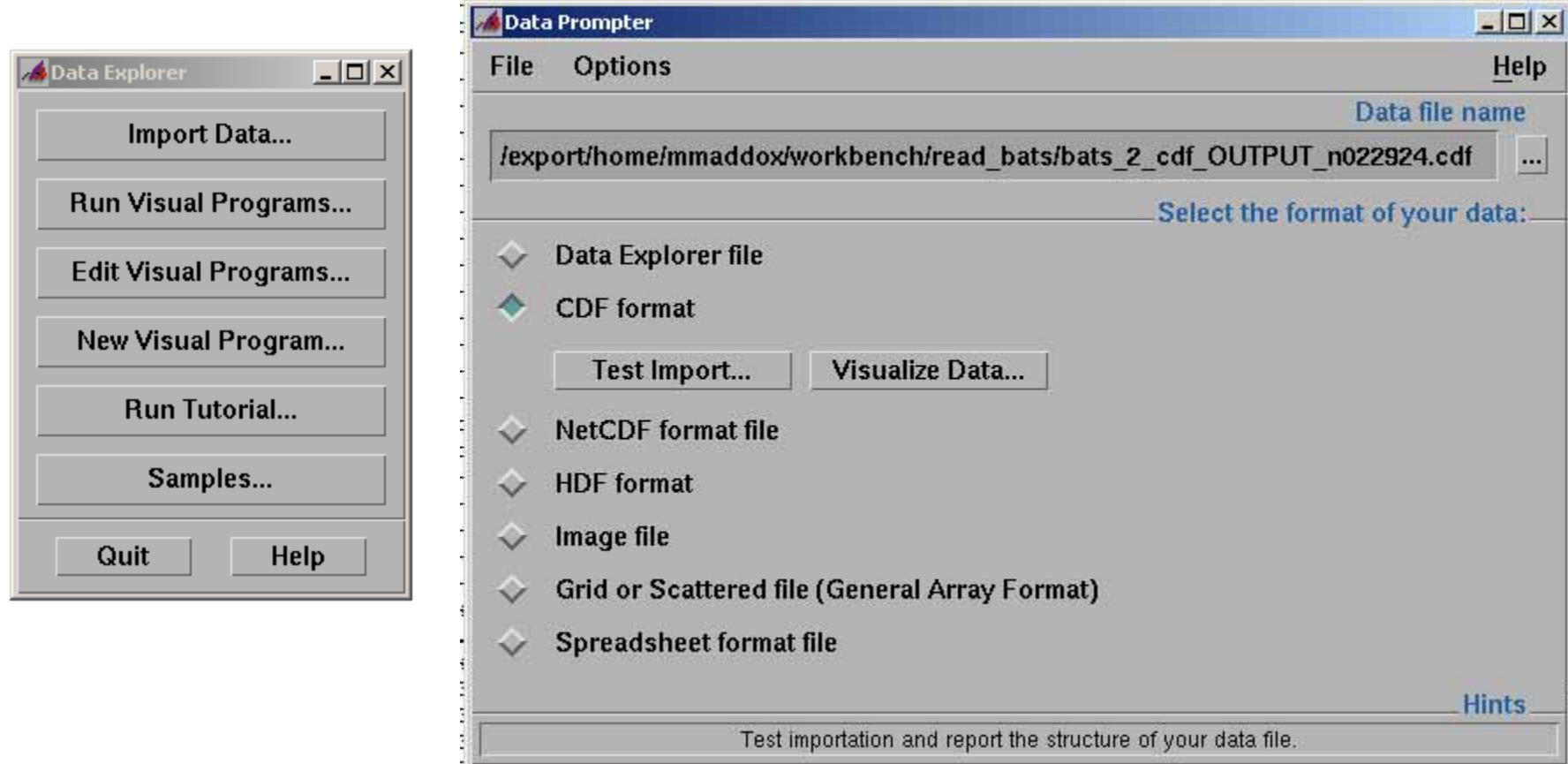
CDF Data Access

- Test CDF data access with an external application
- Use IBM's open data explorer (OpenDX)





Reading CDF Data With OpenDx



Message Window

File Edit Execute Commands Options Help

```
Starting DX executive
Memory cache will use 2047 MB (127 for small items, 1919 for large)
port = 1900
server: accepted connection from client
0: worker here [14173]
Begin Execution
Object Description:
Input object is a Group which contains 18 members.
member 0 is named e
member 1 is named p
member 2 is named rho
member 3 is named jz
member 4 is named jy
...
member 13 is named uy
member 14 is named ux
member 15 is named z
member 16 is named y
member 17 is named x
Each group member is a Field, the basic data carrying structure in DX.
The positions are enclosed within the box defined by the corner points:
[ 0 0 0 ] and [ 1.29341e+06 0 0 ]
Data range is:
minimum = -3.96197e+06, maximum = 3.96197e+06, average = -10.1806
(These are the scalar statistics which will be used by modules
which need scalar values. The length is computed for vectors and
Begin Execution
the determinant for matrices.)
Input is not ready to be rendered because 18 Fields in the Input object do not have colors yet.
Use the 'AutoColor', 'AutoGreyScale', or 'Color' modules to add colors.

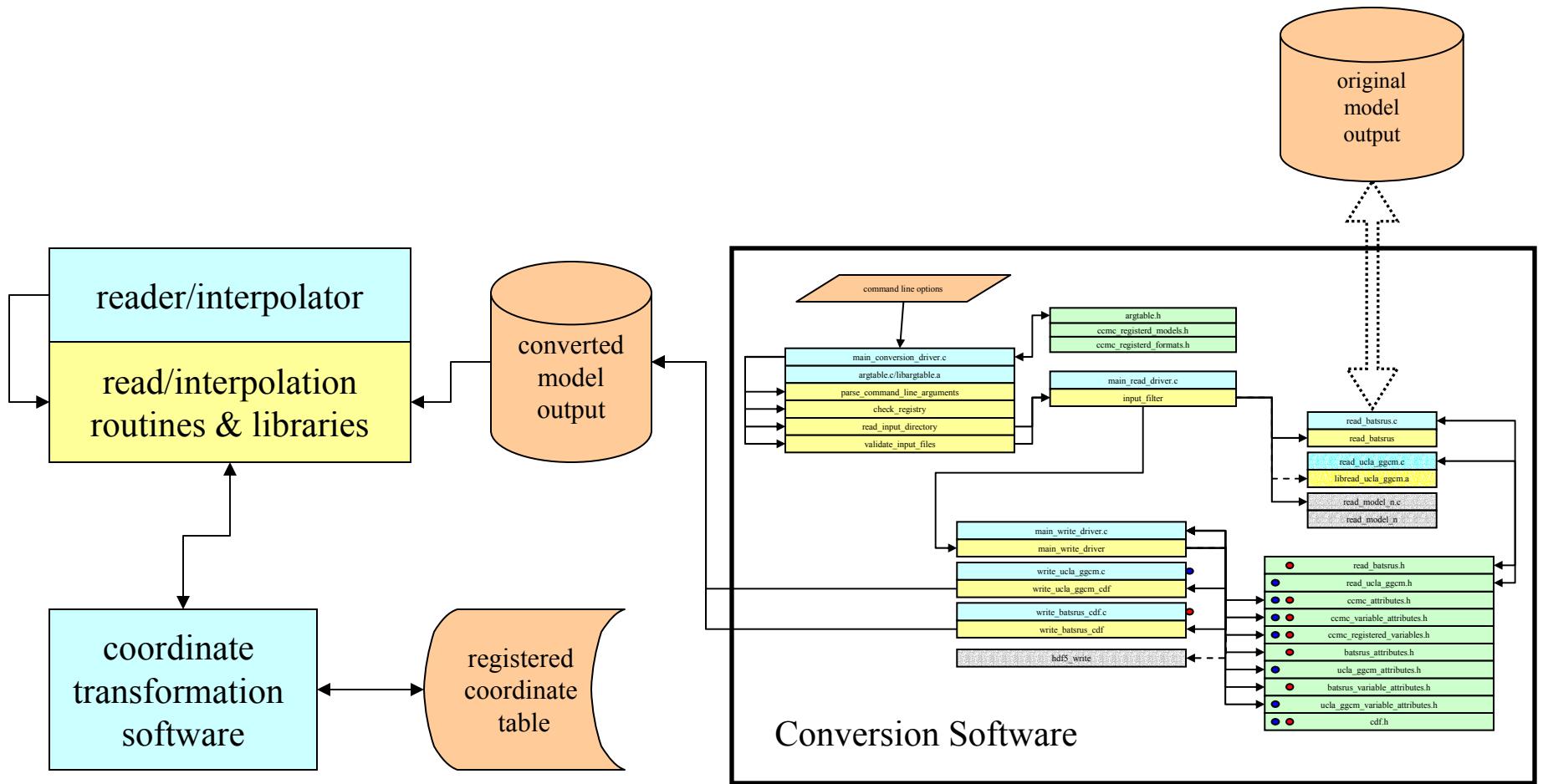
ECHO:
```

OpenDX

Import log

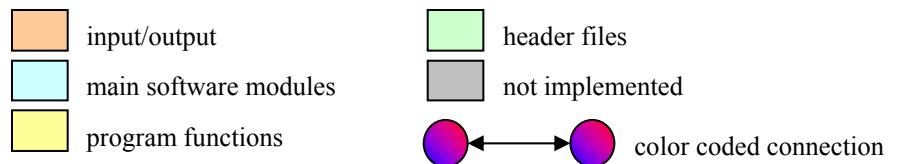
successfully imported all
18 CDF rVariables

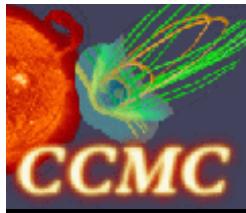
incorrect assumptions
when categorizing data



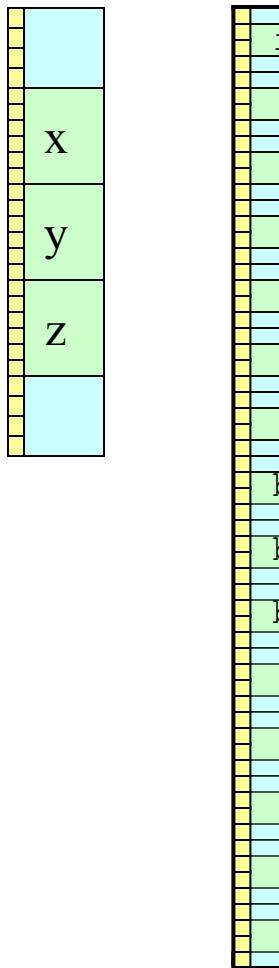
reader/interpolator – built on top of read/interpolation libraries and routines which themselves can be called directly from an external program.

coordinate transformation software – called to convert a user supplied coordinate system to the system in which the model data is currently stored.

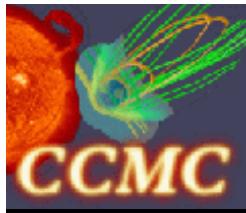




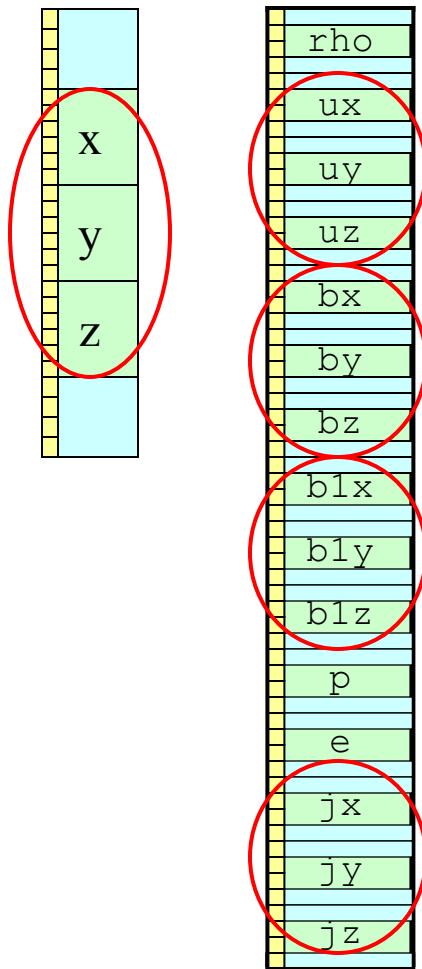
BATSRUS - CDF Variables



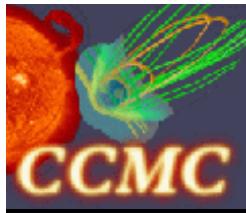
- BATSRUS model contains 18 dynamic variables
 - 3 position variables
 - 15 plot variables
- grouped together variables that had three vector components
 - single variables
 - one dimensional
 - three elements



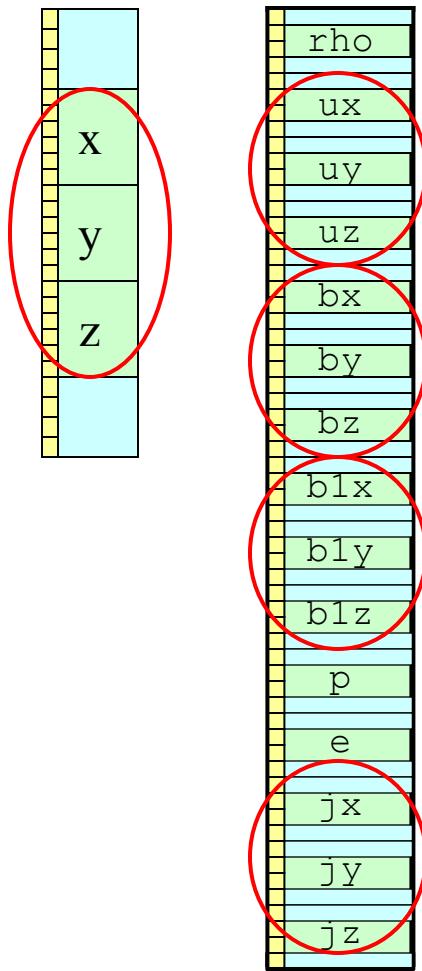
BATSRUS - CDF Variables



- Grouped together variables that had three vector components
 - single variables
 - one dimensional
 - three elements
- cell_position [x,y,z]
- velocity [ux,uy,uz]
- magnetic_field [bx,by,bz]
- deviative_magnetic_field [b1x,b1y,b1z]
- current [jx,jy,jz]



BATSRUS - CDF Variables



- Transformed 15 scalar variables into 5 one-dimensional three-element variables
 - cell_position [x,y,z]
 - velocity [ux,uy,uz]
 - magnetic_field [bx,by,bz]
 - deviative_magnetic_field [b1x,b1y,b1z]
 - current [jx,jy,jz]
- Retained three scalar variable
 - rho
 - p
 - e



BATRUS .OUT to CDF

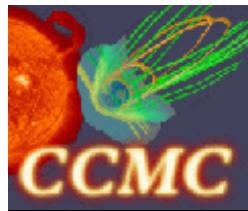
cell_position variable [x, y, z]

R1	1:[1] = -251.0 1:[2] = -44.0 1:[3] = 4.0
R2	2:[1] = -243.0 2:[2] = -44.0 2:[3] = 4.0
R3	3:[1] = -235.0 3:[2] = -44.0 3:[3] = 4.0

R4	4:[1] = -227.0 4:[2] = -44.0 4:[3] = 4.0
R5	5:[1] = -219.0 5:[2] = -44.0 5:[3] = 4.0
R6	6:[1] = -211.0 6:[2] = -44.0 6:[3] = 4.0

R 1,293,406	1293406:[1] = -251.0 1293406:[2] = -36.0 1293406:[3] = 4.0
R 1,293,407	1293407:[1] = -243.0 1293407:[2] = -36.0 1293407:[3] = 4.0
R 1,293,408	1293408:[1] = -235.0 1293408:[2] = -36.0 1283408:[3] = 4.0





BATRUS .OUT to CDF

first column indicates current record number

column two references the current records element index – used for one-dimensional three element vector variables were elements 1, 2, & 3 refers to the x, y, & z components of the vector respectively

R1	1:[1] = -251.0 1:[2] = -44.0 1:[3] = 4.0	R4	4:[1] = -227.0 4:[2] = -44.0 4:[3] = 4.0	R 1,293,406	1293406:[1] = -251.0 1293406:[2] = -36.0 1293406:[3] = 4.0
R2	2:[1] = -243.0 2:[2] = -44.0 2:[3] = 4.0	R5	5:[1] = -219.0 5:[2] = -44.0 5:[3] = 4.0	R 1,293,407	1293407:[1] = -243.0 1293407:[2] = -36.0 1293407:[3] = 4.0
R3	3:[1] = -235.0 3:[2] = -44.0 3:[3] = 4.0	R6	6:[1] = -211.0 6:[2] = -44.0 6:[3] = 4.0	R 1,293,408	1293408:[1] = -235.0 1293408:[2] = -36.0 1283408:[3] = 4.0



CDF Attributes

```
! Skeleton table for the "3d_ful_1_20000101_060000_0083.out.cdf" CDF.
! Generated: Friday, 28-May-2004 16:30:18
! CDF created/modified by CDF V2.7.1
! Skeleton table created by CDF V2.7.1

#header

      CDF NAME: 3d_ful_1_20000101_060000_0083.out.cdf
      DATA ENCODING: NETWORK
      MAJORITY: ROW
      FORMAT: SINGLE

! Variables  G.Attributes  V.Attributes  Records  Dims  Sizes
! -----  -----  -----  -----  -----  -----
    0/39        44          10        0/z        1    18071856

#GLOBALattributes

! Attribute      Entry      Data
! Name          Number      Type      Value
! -----  -----  -----  -----
"README"           1:  CDF_CHAR    { "Model Description:  " -
                           "BATS-R-US, the " -
                           "Block-Adaptive-Tree-Solarw" -
```



CDF Attributes

```
"Run_Registration_number"
 1:  CDF_INT4      { 123456789 } .

"Elapsed_Time_In_Seconds"
 1:  CDF_DOUBLE    { 4200.16 } .

"Number_Of_Dimensions"
 1:  CDF_INT4      { -1073217536 } .

"Number_Of_Special_Parameters"
 1:  CDF_INT4      { 10 } .

"Special_Parameters"
 1:  CDF_REAL8     { 1.666667 }
 2:  CDF_REAL8     { 2248.43 }
 3:  CDF_REAL8     { -0.368162 }
 4:  CDF_INT4      { 3 }
 5:  CDF_INT4      { 1 }
 6:  CDF_INT4      { 1 }
 7:  CDF_INT4      { 3 }
 8:  CDF_INT4      { 6 }
 9:  CDF_INT4      { 6 }
10:  CDF_INT4      { 6 } .

"Number_Of_Plot_Variables"
 1:  CDF_INT4      { 15 } .

"X_Dimension_Size"
 1:  CDF_INT4      { 1293408 } .

"Y_Dimension_Size"
 1:  CDF_INT4      { 1 } .

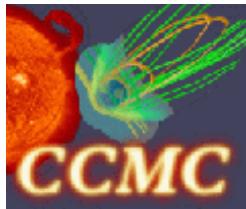
"Z_Dimension_Size"
 1:  CDF_INT4      { 1 } .

"Current_Iteration_Step"
 1:  CDF_INT4      { 22924 } .
```



CDF Variables

! Variable	Data	Number			Record	Dimension
! Name	Type	Elements	Dims	Sizes	Variance	Variances
! -----	----	-----	---	-----	-----	-----
"bx"	CDF_FLOAT	1	1	18071856	T	T
! Attribute	Data					
! Name	Type	Value				
! -----	----	-----				
"min"	CDF_CHAR	{ "-1.0e+009" }				
"max"	CDF_CHAR	{ "1.0e+009" }				
"units"	CDF_CHAR	{ "nT" }				
"grid_system"	CDF_CHAR	{ "grid_system_1" }				
"mask"	CDF_CHAR	{ "UNDEFINED" }				
"description"	CDF_CHAR	{ "X Magnetic Field Component, Earth" - "field up to 60000 nT, Solar field up" - "to 3500 G in sunspots (3.5e8 nT), 1" - "T is not exceeded in" - "Solar/Heliospheric/Earth context" }				
"is_vector_component"	CDF_CHAR	{ "YES" }				
"position_grid_system"	CDF_CHAR	{ "grid_system_1" }				
"data_grid_system"	CDF_CHAR	{ "grid_system_1" } .				



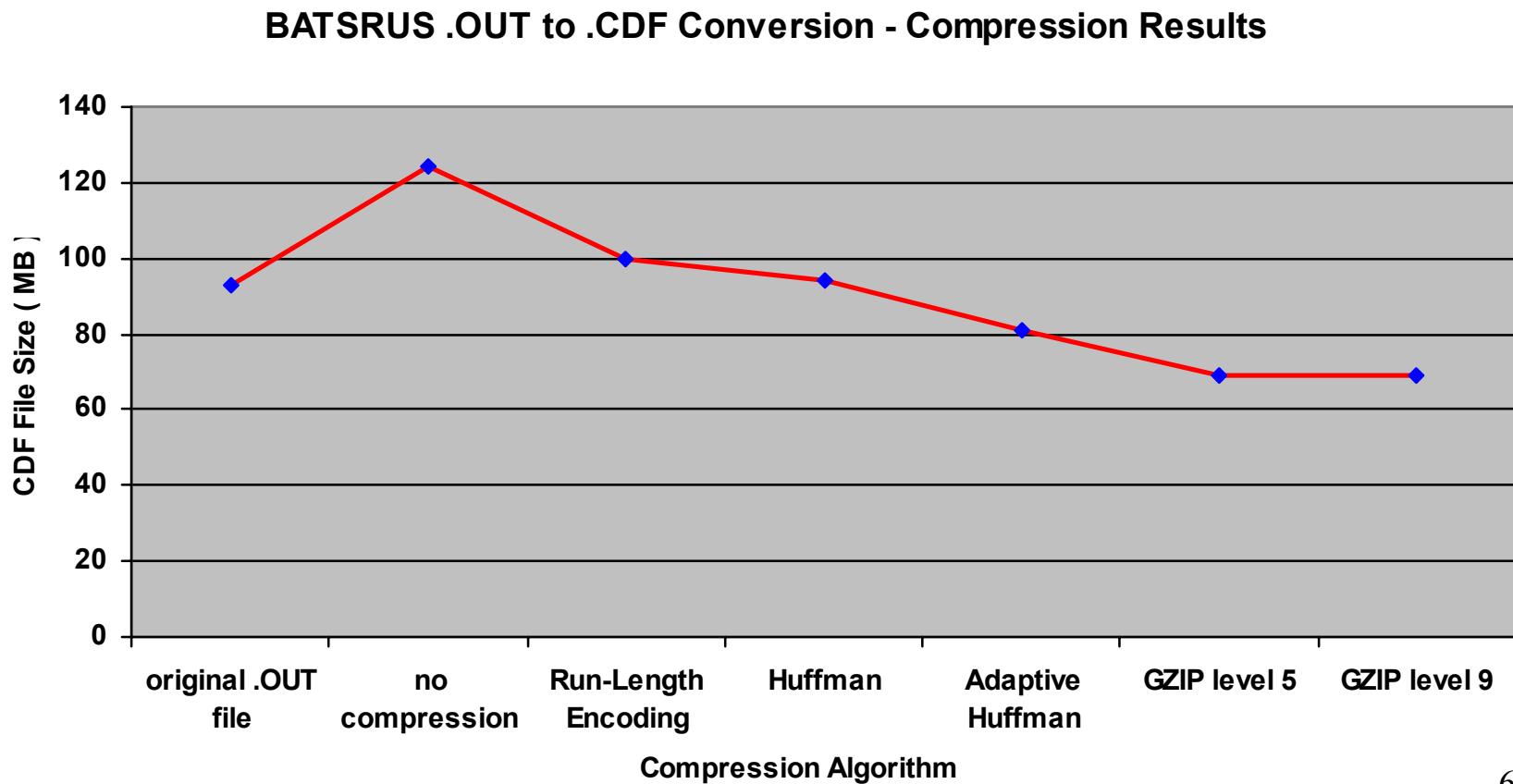
BATRUS .OUT to CDF

- Original BATRUS .OUT file
 - 92 MB
 - 1,293,408 grid positions
- BATRUS CDF Output File
 - 124 MB
 - 10,347,264 records
 - 6,467,040 three-element vector variable records
 - 3,880,224 scalar variable records



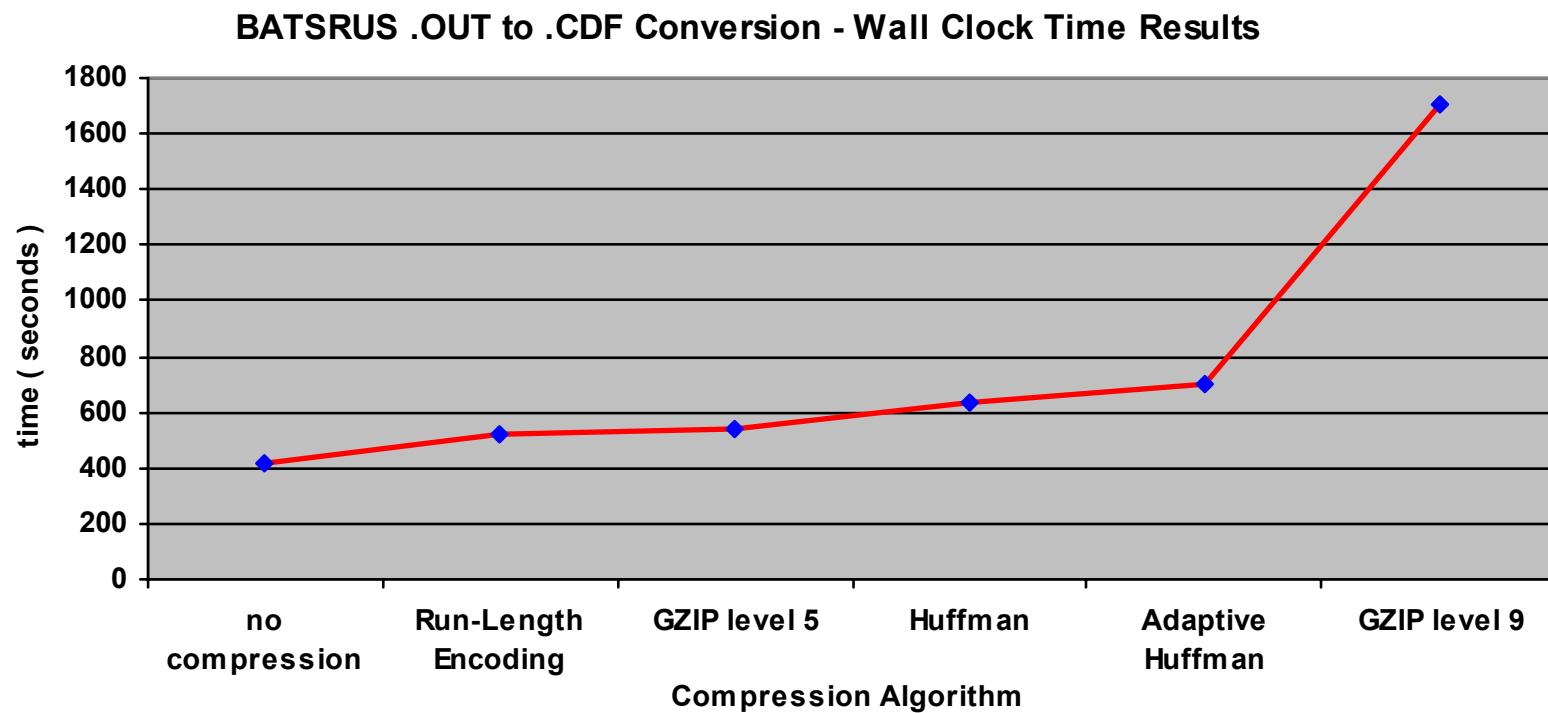


Compression Performance Tests





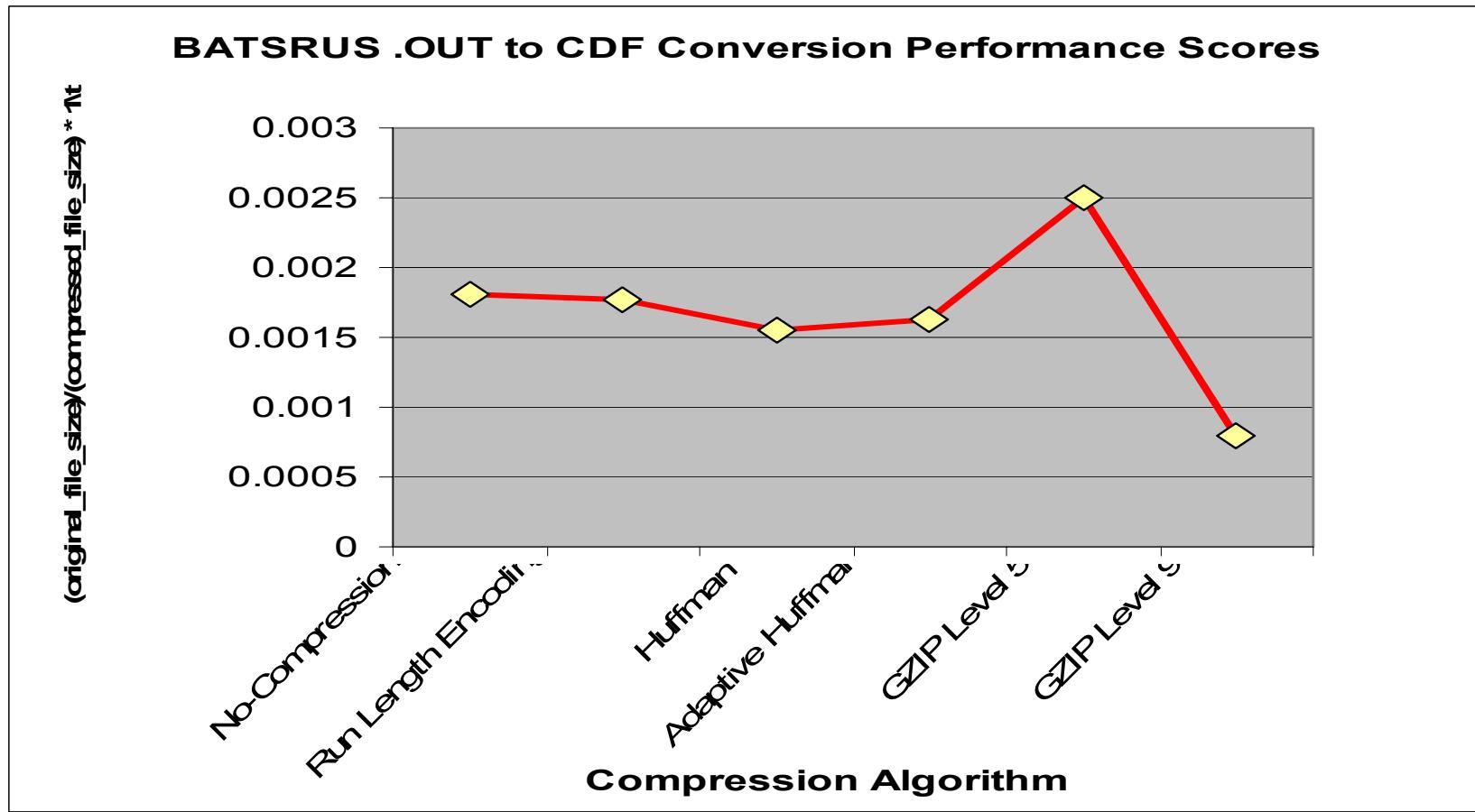
Compression Performance Tests





Performance Score

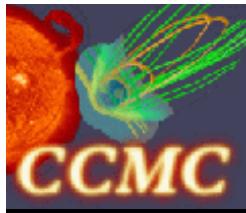
$$\frac{(\text{original_file_size})}{(\text{cdf_file_size})} * \frac{1}{t}$$



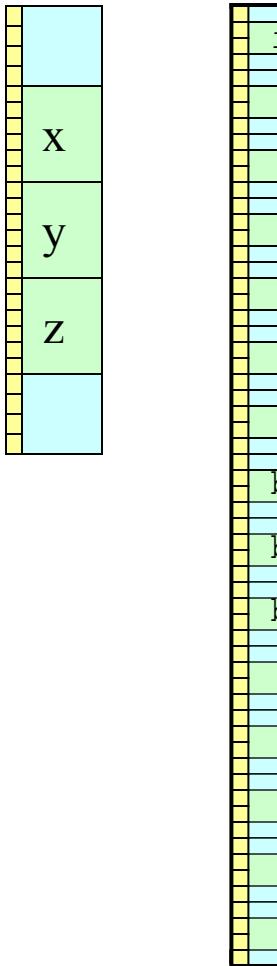


Redesign the CDF Storage Scheme

- Multiple records for every variable value proved unacceptable
- Change CDF to store each BATSRUS variable as
 - Single record CDF rVariable
 - Dimension size equal to number of grid positions



CCMC CDF Variables



- BATSRUS model contains 18 dynamic variables
 - 3 position variables
 - 15 plot variables
- ~~grouped together variables that had three vector components~~
 - single variables
 - one dimensional
 - three elements



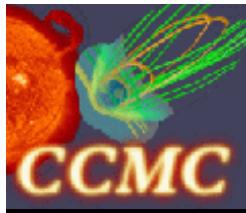
```
1301174030      3d_ful_1_20000101_060000_0083.out
```

```
1308866972      3d_ful_1_20000101_060000_0083.out.cdf
```

```
real        4:30.75
```

```
user        44.73
```

```
sys        53.39
```



Conversion Software Options

```
kauai.ccmc.gsfc.nasa.gov:mmaddox>./ccmc_converter -help
```

```
Usage: ./ccmc_converter [-indir Input Dir] [-outdir Output Dir] [-aux_file  
<grid_file>] -model <batsrus> -format <cdf> [-verbose<on/off>] [-version] [-help]  
[file_name_1 ... file_name_10]
```

-indir Input Dir	Directory Containing Files To Be Converted
-outdir Output Dir	Output Directory
-aux_file <grid_file>	Auxiliary Input File ie. a general Grid File seperate from data files
-model <batsrus>	Model Name
-format <cdf>	Data Format to Convert to
-verbose<on/off>	Verbose Output
-version	Print Version
-help	Print Help/Usage
file_name_1 ... file_name_10	NOTE: ONLY 10 input files can be specified from command line! Use -indir option to use more.

```
kauai.ccmc.gsfc.nasa.gov:mmaddox>
```

CCMC Global Attributes

Attribute	Description	Status
README	General description describing the model, contents of CDF, and HOWTO usage.	
README_visualization	Guidelines for visualizing data contained in file.	
model_name	Name of the registered model that produced the data	
model_type	The type of model used to produce data i.e. Global Magnetosphere	
generation_date	Date of generation or run date...	
original_output_file_name	Name of the original model data file that was converted to current CDF file	
run_registration_number	CCMC Runs on Request registration number for runs submitted through online system	
generated_by	Personal identifying info (First Name Last Name)	
terms_of_usage	<i>For tracking purposes for our government sponsors, we ask that you notify the CCMC whenever you use CCMC results in a scientific publication or presentation.</i>	
grid_system_count	The number n of how many grid systems are used and/or described in the current cdf file NOTE: If n > 1 the additional grid attributes will be defined in corresponding model_attributes.h file	
grid_system_n_number_of_dimensions	The number m of how many dimensions are in grid n. So for every grid there will be a corresponding grid_system_n_number_of_dimensions attribute i.e. The first grid will have an attribute grid_system_1_number_of_dimensions	
grid_system_n_dimension_m_size	Size of dimension m for grid n	
grid_system_n	Outline how particular grid system is defined by showing coordinates used ie. [X, Y, Z] were X,Y,Z are position variables defined in current CDF.	
output_type	Define the type of output is contained in CDF file. i.e. Global Magnetosphere model with Ionosphere output	
standard_grid_target	Defines a standard target grid and coordinate system for which the current models output can be converted to using an external coordinate transformation package.	
grid_n_type	Keywords identifying all grids used in current model output. Grid types will be registered in external coordinate transformation package.	



CCMC Global Attributes

Attribute	Description	Status
README	General description describing the model, contents of CDF, and HOWTO usage.	
README_visualization	Guidelines for visualizing data contained in file.	
model_name	Name of the registered model that produced the data	
model_type	The type of model used to produce data i.e. Global Magnetosphere	
generation_date	Date of generation or run date...	
original_output_file_name	Name of the original model data file that was converted to current CDF file	
run_registration_number	CCMC Runs on Request registration number for runs submitted through online system	
generated_by	Personal identifying info (First Name Last Name)	
terms_of_usage	<i>For tracking purposes for our government sponsors, we ask that you notify the CCMC whenever you use CCMC results in a scientific publication or presentation.</i>	
grid_system_count	The number n of how many grid systems are used and/or described in the current cdf file NOTE: If n > 1 the additional grid attributes will be defined in corresponding model_attributes.h file	
grid_system_n_number_of_dimensions	The number m of how many dimensions are in grid n. So for every grid there will be a corresponding grid_system_n_number_of_dimensions attribute i.e. The first grid will have an attribute grid_system_1_number_of_dimensions	
grid_system_n_dimension_m_size	Size of dimension m for grid n	
grid_system_n	Outline how particular grid system is defined by showing coordinates used ie. [X, Y, Z] were X,Y,Z are position variables defined in current CDF.	
output_type	Define the type of output is contained in CDF file. i.e. Global Magnetosphere model with Ionosphere output	
standard_grid_target	Defines a standard target grid and coordinate system for which the current models output can be converted to using an external coordinate transformation package.	
grid_n_type	Keywords identifying all grids used in current model output. Grid types will be registered in external coordinate transformation package.	



CCMC Global Attributes

Attribute	Description	Status
README	General description describing the model, contents of CDF, and HOWTO usage.	
README_visualization	Guidelines for visualizing data contained in file.	
model_name	Name of the registered model that produced the data	
model_type	The type of model used to produce data i.e. Global Magnetosphere	
generation_date	Date of generation or run date...	
original_output_file_name	Name of the original model data file that was converted to current CDF file	
run_registration_number	CCMC Runs on Request registration number for runs submitted through online system	
generated_by	Personal identifying info (First Name Last Name)	
terms_of_usage	<i>For tracking purposes for our government sponsors, we ask that you notify the CCMC whenever you use CCMC results in a scientific publication or presentation.</i>	
grid_system_count	The number n of how many grid systems are used and/or described in the current cdf file NOTE: If n > 1 the additional grid attributes will be defined in corresponding model_attributes.h file	
grid_system_n_number_of_dimensions	The number m of how many dimensions are in grid n. So for every grid there will be a corresponding grid_system_n_number_of_dimensions attribute i.e. The first grid will have an attribute grid_system_1_number_of_dimensions	
grid_system_n_dimension_m_size	Size of dimension m for grid n	
grid_system_n	Outline how particular grid system is defined by showing coordinates used ie. [X, Y, Z] were X,Y,Z are position variables defined in current CDF.	
output_type	Define the type of output is contained in CDF file. i.e. Global Magnetosphere model with Ionosphere output	
standard_grid_target	Defines a standard target grid and coordinate system for which the current models output can be converted to using an external coordinate transformation package.	
grid_n_type	Keywords identifying all grids used in current model output. Grid types will be registered in external coordinate transformation package.	

